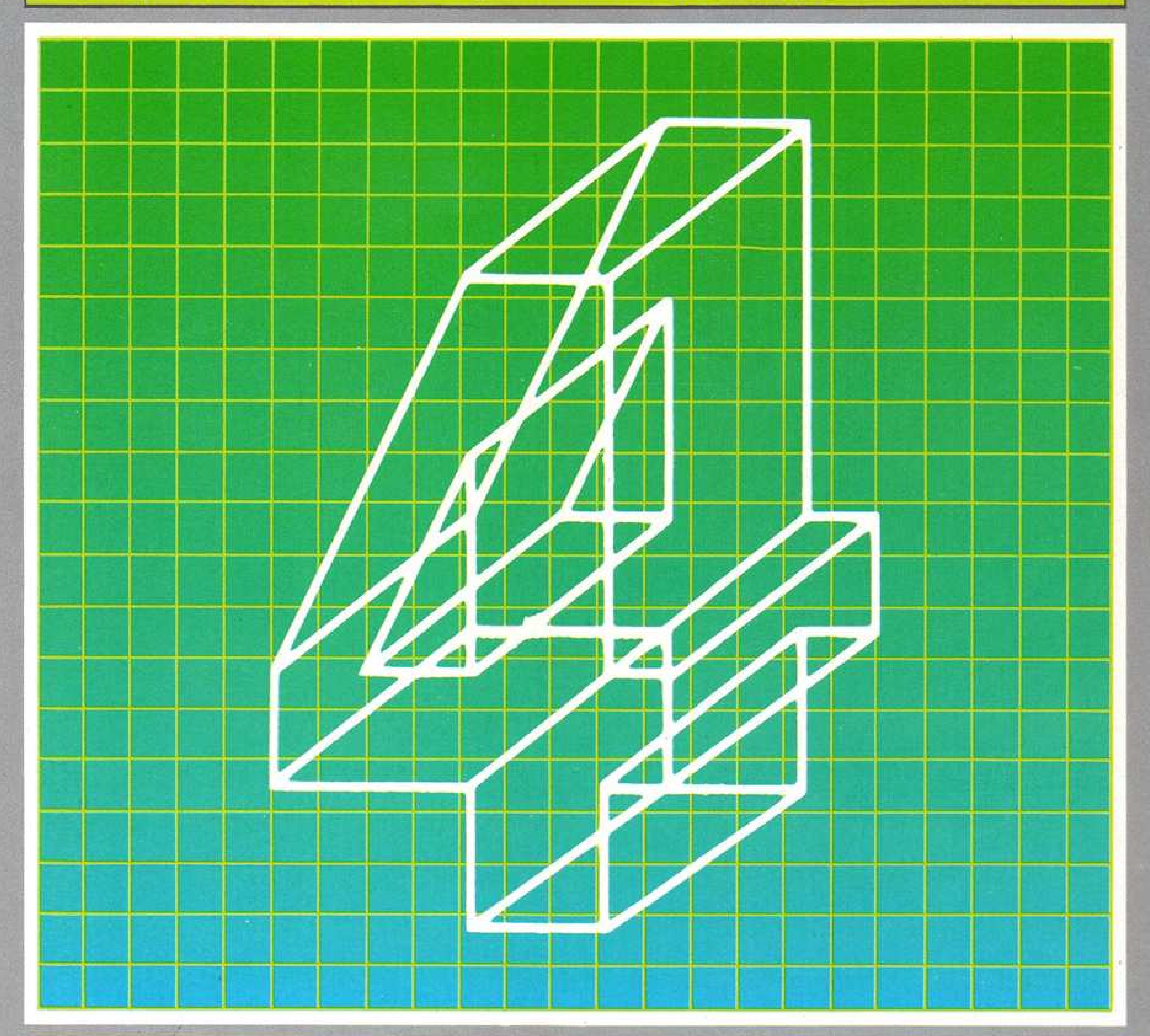
BIBLIOTECA PRACTICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

TALLER DE INFORMATICA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



DIVERTIDOS PROGRAMAS DE BOTANICA, LITERATURA, FISICA, ETC.

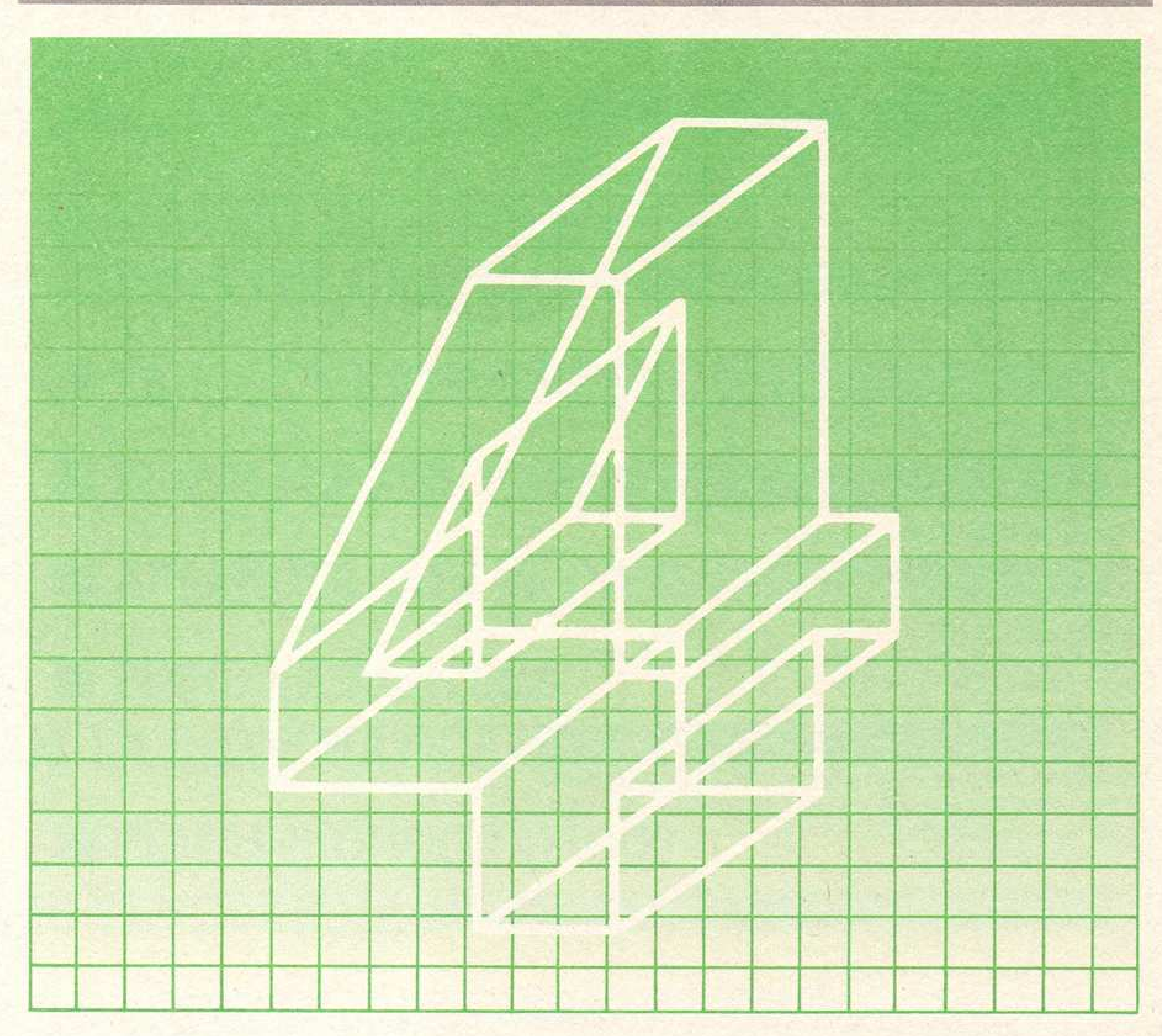
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:
RICARDO ESPAÑOL CRESPO.

Gerente:
ANTONIO G. CUERPO.

Directora de producción:
MARIA LUISA SUAREZ PEREZ.

Director de la colección:
JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:
BRAVO-LOFISH.

Maquetación:
D. SIMON

Dibujos:
JOSE OCHOA

Tomo 4. «Experiencias prácticas en logo», Carlos Albert, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprender con el ordenador», AULA DE INFORMATICA APLICADA: Fernando Suero. Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química; Francisco Blanca, Diplomado en Telecomunicación; María José Hernando, Diplomada en Informática; Soledad Tamariz, Diplomada en Telecomunicación. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:
Sor Angela de la Cruz, 24-7.° G. Teléf. 279 40 36. 28020 Madrid.

Publicidad:
Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:
COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.
Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:
CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.
Buenos Aires - 1.290. Argentina.

ISBN del tomo: 84-7688-054-5
ISBN de la obra: 84-7688-047-2

Fotocomposición:
ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:
MATEU CROMO. PINTO (Madrid).

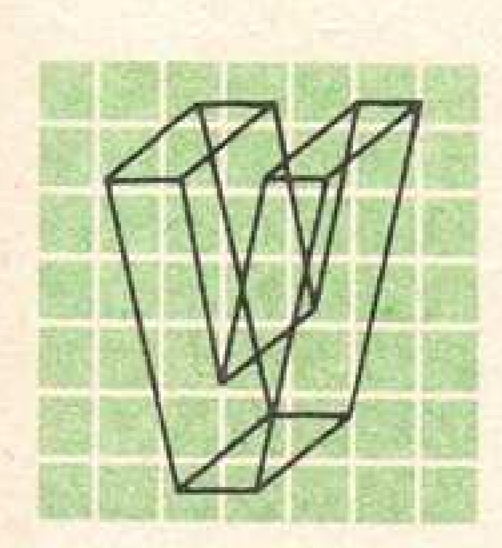
Depósito legal: M-43.185-1986
Printed in Spain - Impreso en España.

Suscripciones y números atrasados:
Ediciones Siglo Cultural, S. A.
Sor Angela de la Cruz, 24-7.° G. Teléf. 279 40 36. 28020 Madrid.

Enero, 1987.

INDICE

EXPERIENCIA Y PRACTICAS EN LOGO



VAMOS a explicarte cómo puedes cambiar la forma de la tortuga y crear una a tu gusto.

Con la orden:

EDFORMA "(nombre)

te aparece en la pantalla una matriz de 16 × 16 cuadrados y en su interior un punto que puedes ir moviendo en todas direcciones con las teclas de desplazamiento del cursor.

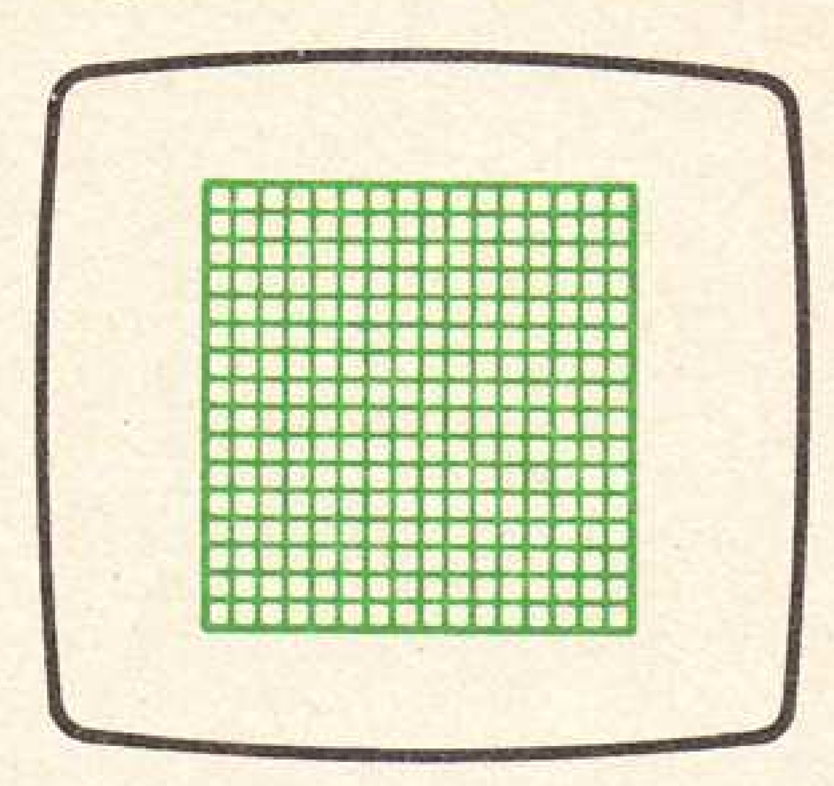


Fig. 1.

Cada cuadrado corresponde a un punto de la forma de la tortuga.

Desplázate a lo largo de la matriz y ve dando forma a tu nueva tortuga, seleccionando y rellenando los cuadrados que quieres que la configuren. La forma de rellenarlos es muy sencilla. Una vez situado en un cuadrado determinado, pulsa la tecla R y éste se rellenará. De esta forma ya tienes un punto definido.

Cuando hayas terminado, lo único que tienes que hacer es guardar la nueva forma de la tortuga. Para ello pulsa la tecla F1.

Puedes dar color a la tortuga con sólo pulsar las teclas, 0, 1, 2 ó 3 antes de pulsar la tecla R. Si quieres borrar uno de los cuadrados que hayas rellenado, hazlo dándole el color de fondo de la pantalla.

Durante todo este proceso de creación de tu nueva forma puedes ir viendo en el tamaño real la forma que te va quedando, ya que aparece al lado de la matriz.

También puedes anular la figura diseñada pulsando simultáneamente la tecla ALT y F2.

Esta forma de variar el aspecto de la tortuga cambia en algunos puntos, si no posees un compatible PC, que es para el que te lo hemos explicado.

Si tienes un MSX los cambios que tienes que realizar son los siguientes:

— La orden EDFORMA no admite un nombre y, por tanto, debes dar un número: EDFORMA n.

— En lugar de un punto dentro de la matriz te aparece una cruz.

— Para rellenar un cuadrado pulsa la tecla de espacio. Esta tecla también te permite borrar un cuadrado con sólo pulsarla de nuevo.

— Para anular una figura diseñada tienes que pulsar la tecla F5.

Para estos ordenadores también tienes la posibilidad de las siguientes opciones:

— Si pulsas simultáneamente las teclas CTRL y K o CTRL y HOME, la forma se borra completamente.

— Si pulsas CTRL y H, se borra la línea horizontal a partir de donde esté la cruz.

En el Logo puedo diseñar mi propia tortuga con la forma que quiera.

— Si pulsas CTRL y V, se borra la línea vertical a partir de donde esté la cruz.

— Si pulsas CTRL y R, se restaura la forma inicial y si ha habido alguna modificación la suprime.

Si haces todo lo que te hemos explicado puedes diseñar una forma para la tortuga como ésta:

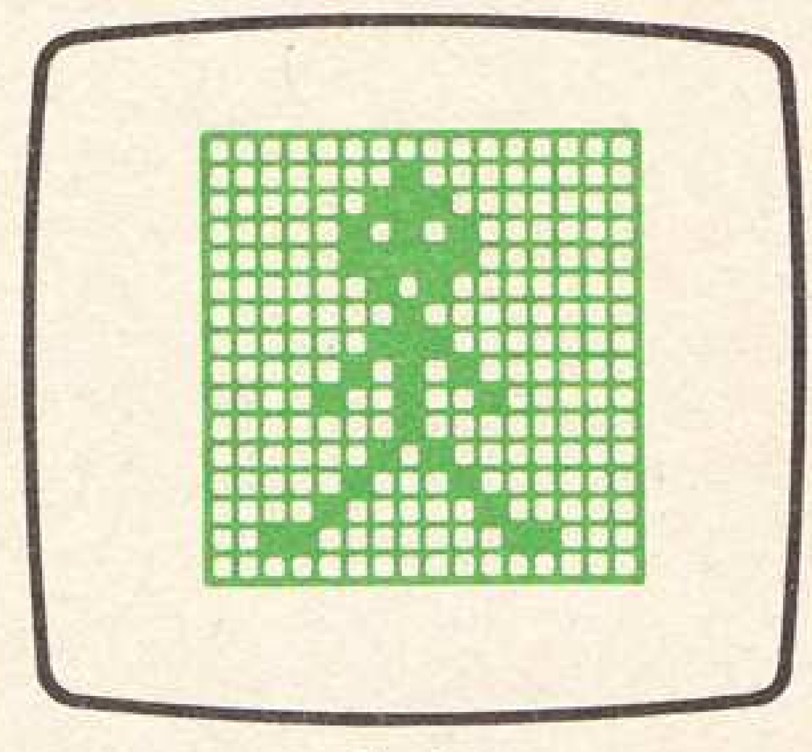


Fig. 2.

Si a partir de este momento quieres trabajar con la tortuga que has creado debes dar la orden:

PONFORMA "nombre

o bien

PONFORMA n

Tanto nombre como n tienen que ser el mismo que hayas dado en la orden EDFORMA.

Aquí tienes las órdenes necesarias para dibujar un simpático animal:

INICIALIZACION

```
? PM
? SL
? BP
? OT
```

CENTRANDO DIBUJO

```
? AV 30 GI 90
? AV 30 GD 90
```

DIBUJO NARIZ

```
? BL
? AV 2 GD 90
? AV 1 GD 90
? AV 2
```

DIBUJO CABEZA

```
? REPITE 12 [AV 9 RE 9 GI 90 AV 1 GD 90]
```

DIBUJO OREJA

```
? RE 5 GD 90
? AV 2 RE 2
? GI 90
```

DIBUJO CUELLO

```
? AV 25
```

DIBUJO CUERPO

```
? REPITE 30 [AV 12 RE 12 GI 90 AV 1
GD 90]
```

DIBUJO RABO

```
? RE 10 REPITE 2 [GI 90 AV 1 GD 90 AV
2 RE 2]
```

DIBUJO PATAS DELANTERAS

```
? SL CENTRO BL
? GI 90 AV 10
? GI 90 AV 20
? GD 90 AV 4
? GD 90 SL
? AV 20 BL
? RE 14 GD 90
? AV 8 GD 90
? AV 4
```

DIBUJO PATAS TRASERAS

```
? SL GI 90
? AV 8 BL
? AV 4 GI 90
? AV 20 GD 90
? AV 4 GD 90
? AV 22 GD 90
? AV 44
```

Como puedes observar lo dibujamos con órdenes que conoces suficientemente. Comenzamos a dibujarlo empezando por la nariz y terminamos por las patas traseras. Lo único que hacemos es dibujar a base de líneas todas las partes del cuerpo.

Si has introducido las órdenes correctamente habrás obtenido un perro como este:

También puedo situar a la tortuga en cualquier parte de la pantalla con las órdenes PONX y PONY.

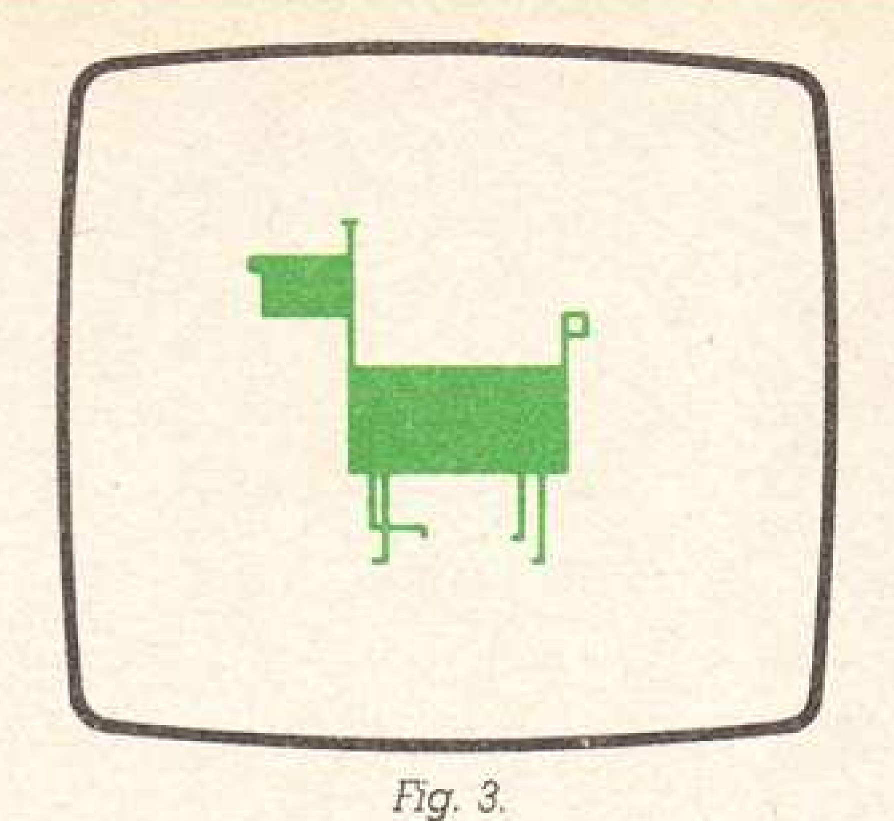


Fig. 3.

Las siguientes órdenes nos dibujan un pequeño laberinto donde viene indicada una entrada y una salida. Te proponemos que con la tortuga, que situarás en la entrada, consigas llegar al final.

DIBUJO LABERINTO
INICIALIZACION

```
? PM
? BP
? SL
? OT
? PONCL 9
```

CENTRANDO DIBUJO

```
? GI 90 AV 45 GD 90
? AV 30 GD 90 BL
```

DIBUJO

```
? AV 65 RE 20 GD 90
? AV 20 GI 90 AV 20
? RE 5 GD 90 AV 25
? GI 90 AV 5 GD 90
? AV 4 GD 90 AV 40
? RE 15 GI 90 AV 20
? SL CENTRO
? GI 90 AV 45 GD 90
? AV 20 GD 90 BL
? AV 30 RE 8 GD 90
? AV 10 RE 5 GI 90
? AV 15 GI 90 AV 10
? RE 10 GD 90 RE 3
? GD 90 AV 15 GD 90
? AV 33 RE 28 GI 90
? AV 5 GI 90 AV 15
? RE 5 GI 90 AV 8
? GD 90 AV 12 GD 90
? AV 15 GD 90 AV 50
? GD 90 RE 14 GD 90
? AV 32 GD 90 AV 12
```

Este dibujo lo realizamos haciendo una serie de líneas que se unen en unos puntos determinados y que van creando una serie de caminos.

Como siempre hacemos al principio, inicializamos el estado, tanto de la tortuga como de la pantalla, y en este caso damos color al lápiz de la tortuga. Luego centramos el dibujo y comenzamos a pintarlo.

Las órdenes que vienen a continuación nos pintan dos flechas indicando la entrada y la salida:

ENTRADA

```
? SL CENTRO
? PONCL 3
? GI 90 AV 65
? GD 90 AV 25
? BL GD 90
? AV 15 GI 135
? AV 5 SL
? RE 5 BL
? GI 90 AV 5
```

SALIDA

```
? SL CENTRO
? RE 45 GI 90
? AV 8 GI 90
? BL AV 15
? GD 135 AV 5
? SL RE 5
? GI 270 BL
? AV 5 RE 5
```

Ya tienes el laberinto dibujado. Te habrá quedado de la siguiente forma:

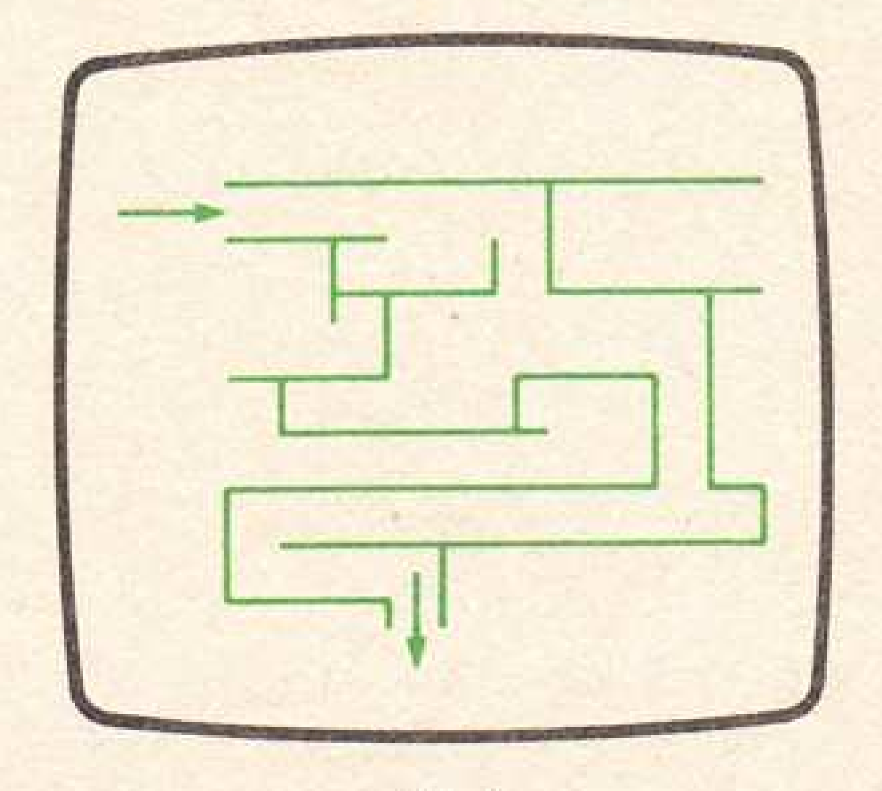


Fig. 4.

Tengo un montón de colores tanto para el fondo de la pantalla como para el color del lápiz.

Ahora sitúa la tortuga en la entrada y condúcela hacia la salida.

Aquí tienes la forma de hacerlo:

SOLUCION

COLOCACION DE LA TORTUGA EN LA SALIDA

```
? SL CENTRO
? PONCL 9
? AV 27 GD 90
? RE 45 MT
? ESPERA 100
```

GUIANDO A LA TORTUGA

```
? BL
? AV 41 ESPERA 20
? GD 90 AV 20 ESPERA 20
? GI 90 AV 15 ESPERA 20
? GD 90 AV 22 ESPERA 20
? GD 90 AV 40 ESPERA 20
? GI 90 AV 8 ESPERA 20
? GI 90 AV 21 ESPERA 20
? GD 90 AV 15
```

No ha sido muy complicado, ¿verdad? Observa que cada vez que la tortuga recorre un cierto tramo hace una pausa; esto se consigue con la orden:

ESPERA n

Ya veremos más adelante esta orden con más detenimiento.

Aquí tienes un sencillo programa que realiza un dibujo abstracto y que nunca se puede repetir exactamente igual.

CUADRO ABSTRACTO
INICIALIZACION

```
? BP
? PM
? OT
? BL
? PONFONDO 1
```

DIBUJO

```
? REPITE 100 [PONCL AZAR 15 AV AZAR 156 GD AZAR 360]
```

Obtendrás un dibujo compuesto por un montón de rayas en todas direcciones y cada una de un color. Todo sobre un fondo negro.

Observa que en este caso hemos introducido una orden nueva. En la orden REPITE le decimos que ponga color al lápiz de la tortuga, que avance y que gire. Pero en este caso no le damos un valor en concreto, como siempre solíamos hacer. Le decimos que ponga un color AZAR 15, que avance AZAR 156 y que gire a la derecha AZAR 360.

¿Qué significa AZAR n? Es muy sencillo. La orden AZAR n nos da un valor totalmente aleatorio comprendido entre 0 y el valor inmediatamente inferior a n. Por ejemplo, si damos la orden:

AZAR 49

nos devuelve un número comprendido entre 0 y 48, ambos inclusive.

Como es lógico, esta orden debe acompañar a alguna otra que precise de un valor numérico, como, por ejemplo, AV o GD.

Así, en nuestro dibujo anterior, obtenemos líneas de diferentes colores, ya que damos la orden PONCL AZAR 15. Cada una tiene un color aleatorio comprendido entre 0 y 14.

De igual forma, cada una es de una longitud, ya que le decimos que avance un número aleatorio entre 0 y 155. Y, por último, cada una está en una dirección, ya que gira, después de pintar una línea, un número de grados aleatorio comprendido entre 0 y 359.

Aquí puedes limitar el tamaño máximo de las líneas con sólo variar el valor de la orden AV. Por ejemplo, si pones la orden:

AV AZAR 10

ninguna línea será mayor de nueve puntos.

Con las siguientes órdenes obtenemos un dibujo que se puede asemejar a una noche oscura con el cielo lleno de estrellas.

CIELO ESTRELLADO
INICIALIZACION

```
? BP
? PM
? PONFONDO 1
? PONCL 15
? OT
```

DIBUJO

```
? REPITE 100 [SL PONX AZAR 256 BL AV 1 SL PONY AZAR 90 BL AV 1]
```

Puedo dejar que el valor numérico que tienen algunas órdenes sea totalmente aleatorio.

Lo primero que hacemos, una vez que hemos inicializado el estado de la tortuga y de la pantalla, es dar color al fondo. En este caso ponemos el fondo de color negro. Luego damos color al lápiz para que nos dibuje las estrellas de color blanco. Una vez determinados los colores, empezamos a dibujar las estrellas. Esto lo hacemos también de una forma aleatoria.

En este caso damos dos órdenes nuevas:

PONX n
PONY n

Estas dos órdenes lo que hacen es desplazar a la tortuga a la posición que determina n en abscisas y ordenadas, respectivamente.

PONX n desplaza a la tortuga horizontalmente hasta el punto de abscisas n.

PONY n desplaza a la tortuga verticalmente hasta el punto de ordenadas n.

Ten en cuenta que la tortuga, cuando está en el centro, está en el punto (0,0), es decir, tanto la abscisa como la ordenada tienen valor cero. Por tanto, hacia la derecha están los valores positivos de la abscisa y hacia la izquierda los negativos. Con las ordenadas pasa lo mismo, sólo que en este caso hacia arriba tenemos los valores positivos y hacia abajo los negativos.

Si dispones de un ordenador con una resolución gráfica de 256 × 192 puntos, el siguiente dibujo te ayudará a verlo:

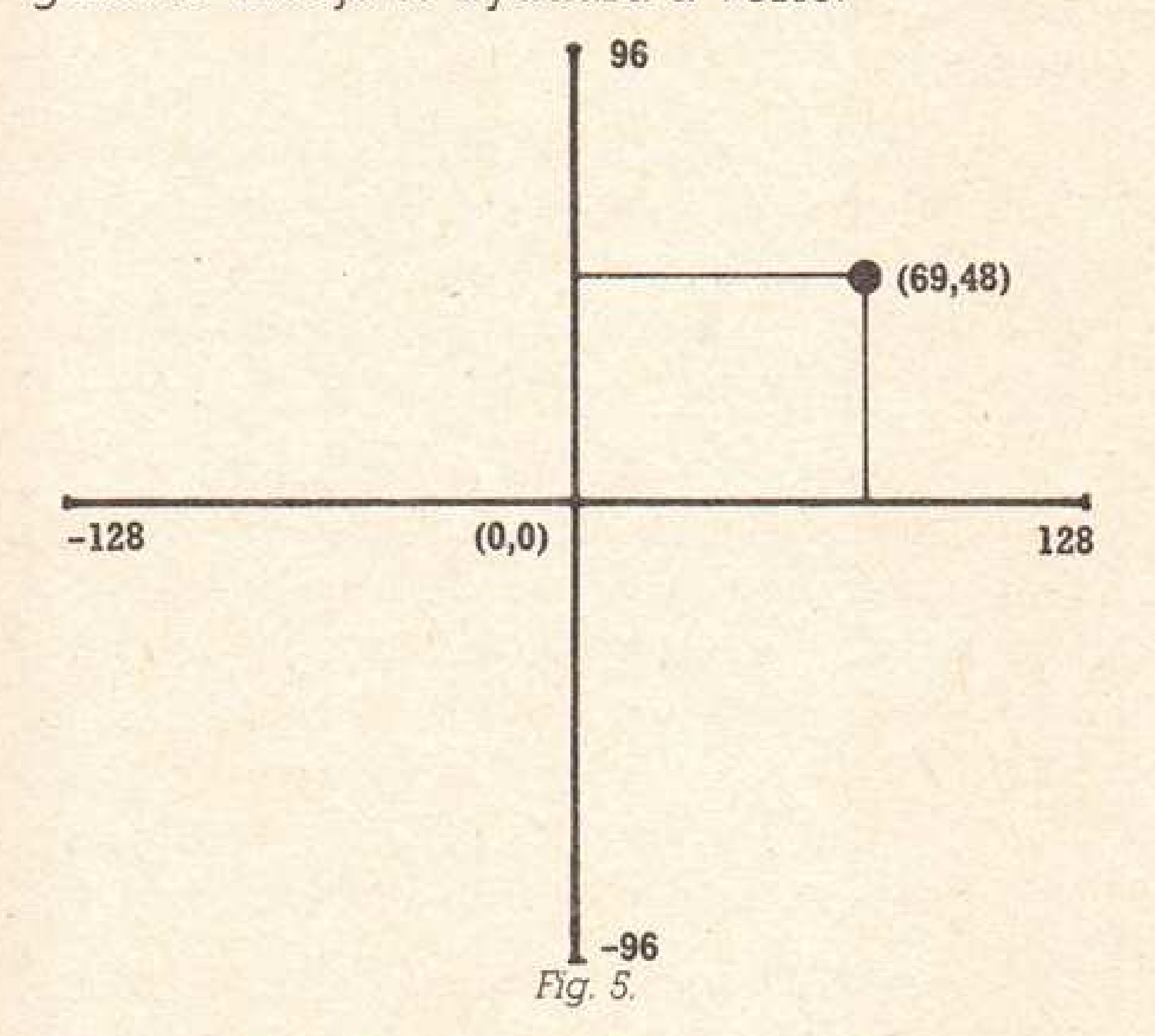


Fig. 5.

Comprueba lo que realizan las siguientes órdenes:

? PM
? MT
? SL
? REPITE 100 [PONX AZAR 128]

La tortuga se desplaza únicamente en la zona que existe desde el centro de la pantalla hasta el extremo derecho de ésta.

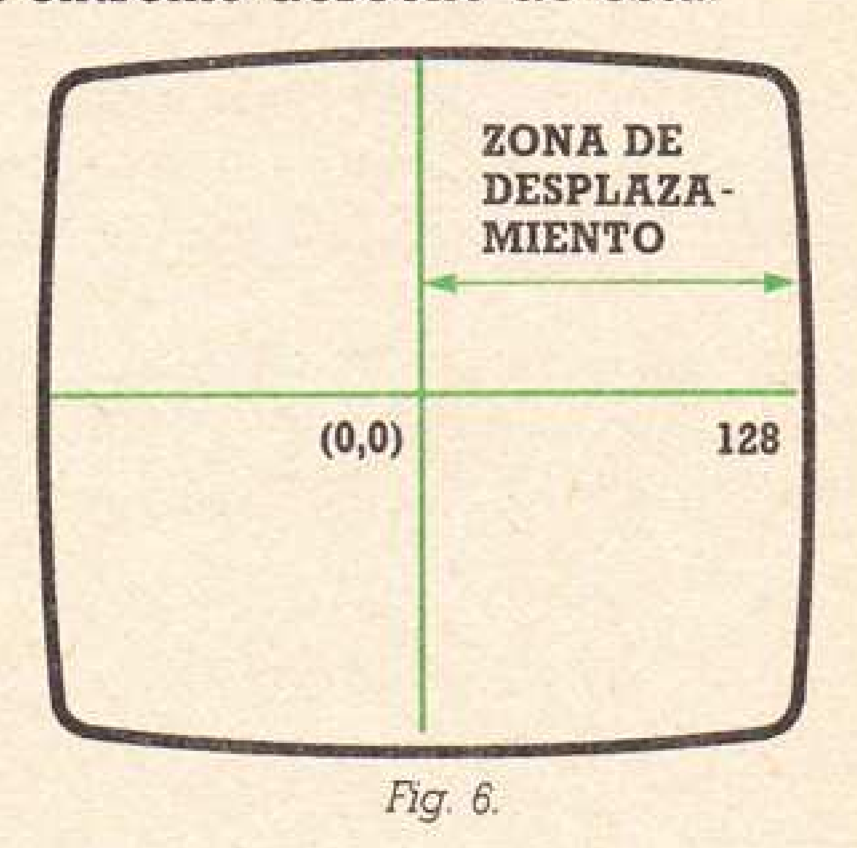


Fig. 6.

Si ahora damos las órdenes:

? PM
? MT
? SL
? REPITE 100 [PONX AZAR 256]

La tortuga se desplaza a lo ancho de toda la pantalla, ya que al darle valores mayores de 128, ésta sale del extremo derecho y aparece por el extremo izquierdo. En este caso todos los valores mayores de 128 y menores de 256 corresponderán a la zona que va desde el centro hasta el extremo izquierdo.

Por tanto, estos valores corresponderán a los comprendidos entre 0 y -128. El 0 corresponde con el 256 y el -128 con el 129.

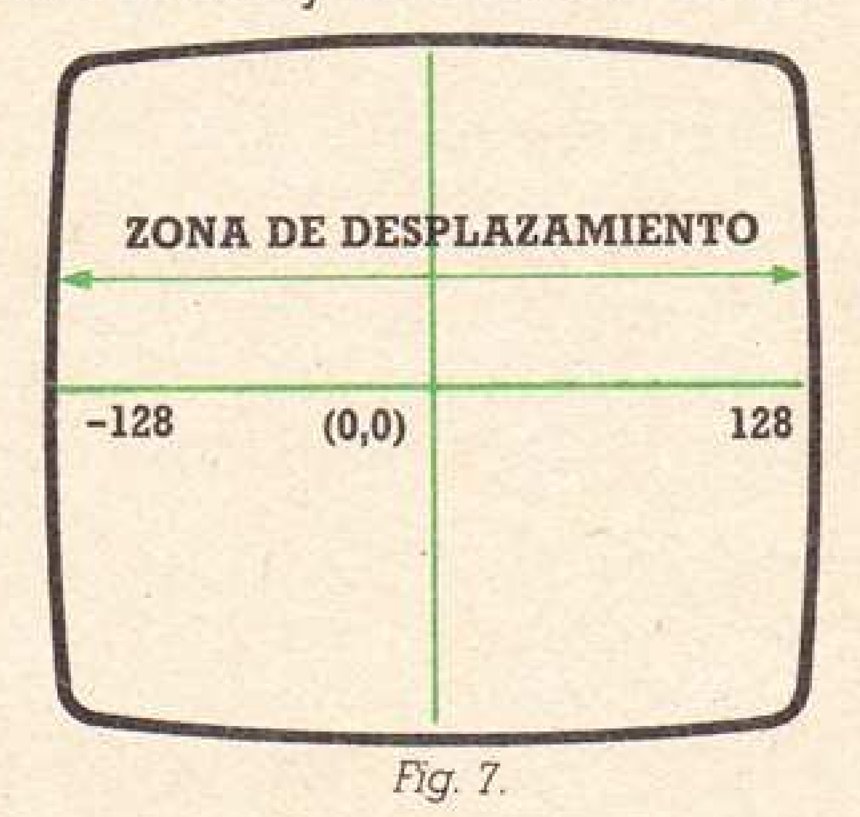


Fig. 7.

Puedo ir realizando pausas entre diferentes órdenes para ver más despacio cómo se van ejecutando.

Volviendo a nuestro dibujo, observa que al dar la orden PONX AZAR 256 conseguimos posicionarnos a lo ancho de toda la pantalla, y con la orden PONY AZAR 90 limitamos una zona que va desde el centro hasta el punto 90 en vertical.

Subimos el lápiz antes de posicionarnos en un punto determinado para que el desplazamiento se realice sin dejar rastro. Una vez situados, bajamos el lápiz y avanzamos un punto. De esta forma vamos consiguiendo una serie de puntos por toda la pantalla de una forma aleatoria y obtenemos algo parecido a un firmamento.

Con las siguientes órdenes podemos obtener una pantalla completamente rayada y así conseguir otro bonito dibujo.

INICIALIZACION

```
? PM
? OT
? BP
? BL
? PONFONDO 4
? PONCL 6
```

DIBUJO

```
? REPITE 100 [PONX AZAR 256 PONY
AZAR 192]
```

En este dibujo hacemos más o menos lo mismo que en el anterior, pero en este caso cuando le ordenamos a la tortuga que se posicione en un punto determinado, lo hace dejando un rastro tras de sí, dibujando, ya que el lápiz lo tiene bajado.

Alternativamente va desplazándose en horizontal y en vertical, obteniendo así un montón de rayas, pero todas ellas rectas.

Por ejemplo, un cuadrado, que siempre lo hacíamos con la orden REPITE, ahora lo podemos hacer también de la siguiente forma:

```
? SL CENTRO BL
? PONY 40 PONX 40
? PONY 0 PONX 0
```

Así obtenemos un cuadrado de 40 puntos de lado.

Colores I

Ya has visto en algunos ejemplos que hemos realizado que con Logo se puede dibujar con colores. Vamos a ver en esta ocasión dos de las órdenes que actúan sobre el color, cómo se dan y qué se obtiene con ellas.

Ordenes que ya hemos utilizado son, por ejemplo:

PONCL y PONFONDO

PONCL es la abreviatura de Pon Color Lápiz. Esta orden debe ir seguida de un número n. Por tanto, su sintaxis es:

PONCL n

El valor de n es el que determina el color que queremos que tenga el lápiz de la tortuga desde ese momento.

Si, por ejemplo, damos la orden:

PONCL 1

todo desplazamiento que se realice desde este momento y siempre que esté el lápiz bajado se hace dejando un rastro tras la tortuga del color que corresponde al valor 1. Dependiendo del ordenador que posea el valor 1 corresponderá a un color determinado.

PONFONDO es la abreviatura de Pon Color Fondo. Al igual que la orden anterior ésta debe ir seguida de un valor n, que es el que nos determina el color que queremos que tenga el fondo de la pantalla, pero sólo de la zona destinada a gráficos.

Todas las versiones del Logo arrancan con un color determinado tanto de fondo como del lápiz de la tortuga. Nosotros podemos, pues, variar estos colores a nuestro gusto, pero tendremos que saber el color que corresponde a cada valor que demos en estas órdenes.

El valor de n, por tanto, está limitado a los colores que posee un determinado ordenador.

A continuación te damos una tabla con los diferentes códigos de colores para tres ordenadores diferentes.

En la posición central, la tortuga está en las coordenadas (0,0).

Antes de nada ten en cuenta lo siguiente:

— En ordenadores como el Spectrum la orden PONFONDO es PONCF.

— En ordenadores compatibles PC tienes la posibilidad de cambiar de grupo de colores, ya que existen dos grupos y cada uno de ellos contiene unos colores determinados. Estos dos grupos son los que se obtienen al dar la orden PALETA 0 y PALETA 1.

Códigos de colores para el fondo

Número	Spectrum Color	MSX Color	Compatibles PC Color
0	Negro	Transparente	Negro
1	Azul	Negro	Azul
2	Rojo	Verde	Verde
3	Magenta	Verde claro	Cyan
4	Verde	Azul oscuro	Rojo
5	Cyan	Azul claro	Morado
6	Amarillo	Rojo oscuro	Marrón
7	Blanco	Azul celeste	Blanco
8	—	Rojo	Gris
9	—	Rojo claro	Azul claro
10	—	Amarillo oscuro	Verde claro
11	—	Amarillo claro	Azul ultramar
12	—	Verde oscuro	Rojo claro
13	—	Magenta	Morado claro
14	—	Gris	Amarillo
15	—	Blanco	Blanco intenso

El color del lápiz coincide en los códigos numéricos con el color del fondo, tanto en el Spectrum como en los MSX. Sólo existen variaciones en los compatibles PC. Este es el cuadro del color para el lápiz en este caso.

Número	Paleta 0 Color	Paleta 1 Color
0	Color del fondo	Color del fondo
1	Cyan	Verde
2	Morado	Rojo
3	Blanco	Amarillo

Las dos paletas no pueden estar activadas a la vez y en el caso que estemos trabajando con una y cambiemos a la otra, los colores se intercambiarán.

Cuadro resumen

— EDFORMA "nombre
EDFORMA n

Permite crear la forma de una nueva tortuga con el nombre o el número especificado en nombre o n, respectivamente. La forma que se puede crear es totalmente libre. Nunca puede exceder de un tamaño de 16 puntos, tanto en vertical como en horizontal.

— PONFORMA "nombre
PONFORMA n

Hace aparecer en la pantalla la forma que se especifica en nombre o n actuando como la tortuga.

— ESPERA n

Detiene la ejecución de lo que se está realizando durante n cincuentésimas de segundo.

— AZAR n

Da un valor entero y positivo aleatorio entre 0 y el valor inmediatamente inferior a n.

— PONX n

Desplaza a la tortuga hasta la posición n en abscisas.

En los ordenadores PC compatibles, cuando damos la orden MT, la pantalla pasa directamente a modo mixto.

— PONY n

Desplaza a la tortuga hasta la posición n en ordenadas.

— PALETA n

Activa el grupo de colores que determina n (0 ó 1).

Ejercicios

1. *Con las órdenes PONX y PONY, haz el siguiente dibujo:*

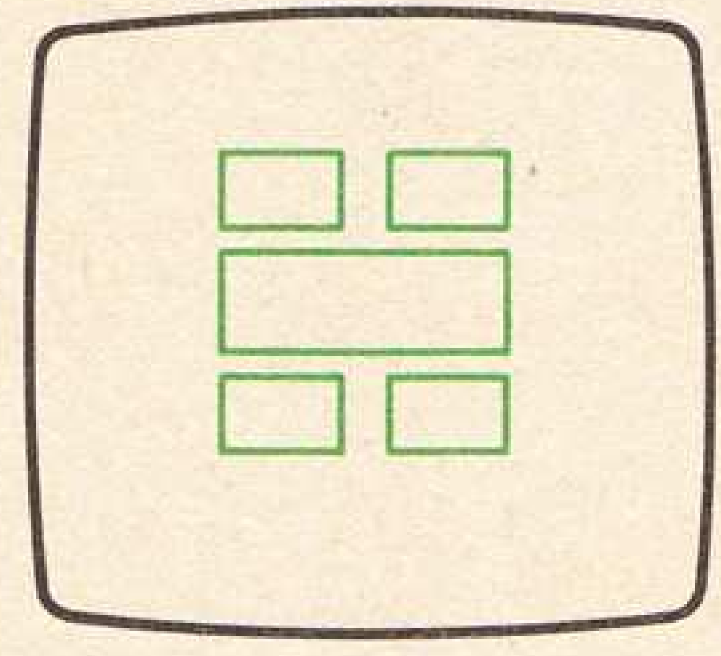


Fig. 8.

2. *Rellena las figuras del dibujo anterior, pero cada una de un color.*
3. *Haz el siguiente dibujo.*

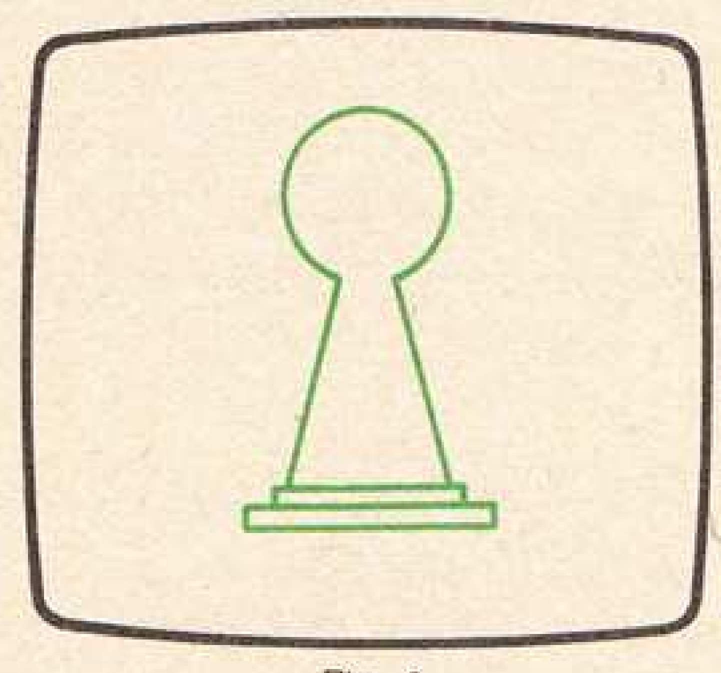


Fig. 9.

4. *Dibuja tres cubos, uno dentro del otro.*
5. *¿Son correctas estas órdenes?*
   - *— PONF*
   - *— AVANZA PONCL 40*
   - *— PONX 600*
   - *— RELLENA 3*
   - *— AV AZAR 300*
   - *— PONFONDO 4 PONCL 3*

Solución a los ejercicios

1:

INICIALIZACION

? PM
? BP
? OT
? SL

CENTRANDO DIBUJO

? GI 90 AV 33 GD 90 RE 5
? BL

DIBUJO CUADROS INFERIORES

? PONY 20 PONX -8
? PONY -5 PONX -33
? SL PONX 7 BL
? PONY 20 PONX 32
? PONY -5 PONX 7

DIBUJO RECTANGULO CENTRAL

? SL PONY 25 PONX 32 BL
? PONY 45 PONX -33
? PONY 25 PONX 32

DIBUJO CUADROS SUPERIORES

? SL PONY 50 BL
? PONY 75 PONX 7
? PONY 50 PONX 32
? SL PONX -8 BL
? PONY 75 PONX -33
? PONY 50 PONX -8

2:

Una vez realizado el dibujo anterior vamos a rellenar las figuras. Sin borrar pantalla ni realizar algún cambio, introduce las órdenes:

COLOCACION INICIAL

? SL CENTRO
? GD 90 AV 15
? GI 90 AV 10

RELLENANDO

? PONCL 1 BL RELLENA
? SL GI 90 AV 25
? PONCL 2 BL RELLENA
? SL GD 90 AV 20
? PONCL 5 BL RELLENA
? SL AV 25
? PONCL 6 BL RELLENA
? SL GD 90 AV 20
? PONCL 7 BL RELLENA

Si los colores que hemos dado no sirven para tu ordenador, cámbialos por los que tenga el tuyo. Te puede servir de ayuda mirar la tabla de código de colores.

Dependiendo del ordenador tengo un número determinado de colores.

3:

INICIALIZACION

```
? PM
? SL
? BP
? OT
```

CENTRANDO DIBUJO

```
? GI 90 AV 35
? GD 90 RE 40
? BL
```

DIBUJO

```
? AV 10 GD 90
? AV 70 GD 90
? AV 10 GD 90
? AV 70 RE 5
? GD 90 SL
? AV 10 BL
? AV 5 GD 90
? AV 60 GD 90
? AV 5 RE 5
? GD 161 AV 70
? GD 90
? REPITE 31 [GI 10 AV 4]
? GD 74 AV 70
```

4:

INICIALIZACION

```
? PM
? BP
? SL
? OT
```

CENTRANDO DIBUJO

```
? RE 40 GD 90
? AV 40 GI 90
? BL
```

PRIMER CUBO

```
? REPITE 4 [AV 80 GI 90]
? GD 45 AV 40 GI 45
? AV 80 GI 135 AV 40
? GD 45 AV 80 GD 135
? AV 40 GD 45 AV 80
? RE 80 GD 90 AV 80
? GI 90 AV 80 RE 80
? GD 135 AV 40
```

SEGUNDO CUBO

```
? SL CENTRO
? RE 30 GD 90
? AV 30 GI 90
? BL
? REPITE 4 [AV 60 GD 90]
? GD 45 AV 30 GI 45
? AV 60 GI 135 AV 30
? GD 45 AV 60 GD 135
? AV 30 GD 45 AV 60
? RE 60 GD 90 AV 60
? GI 90 AV 60 RE 60
? GD 135 AV 30
```

TERCER CUBO

```
? SL CENTRO
? RE 20 GD 90
? AV 20 GI 90
? BL
? REPITE 4 [AV 40 GI 90]
? GD 45 AV 20 GI 45
? AV 40 GI 135 AV 20
? GD 45 AV 40 GD 135
? AV 20 GD 45 AV 40
? RE 40 GD 90 AV 40
? GI 90 AV 40 RE 40
? GD 135 AV 20
```

Te ha tenido que quedar un dibujo como éste:

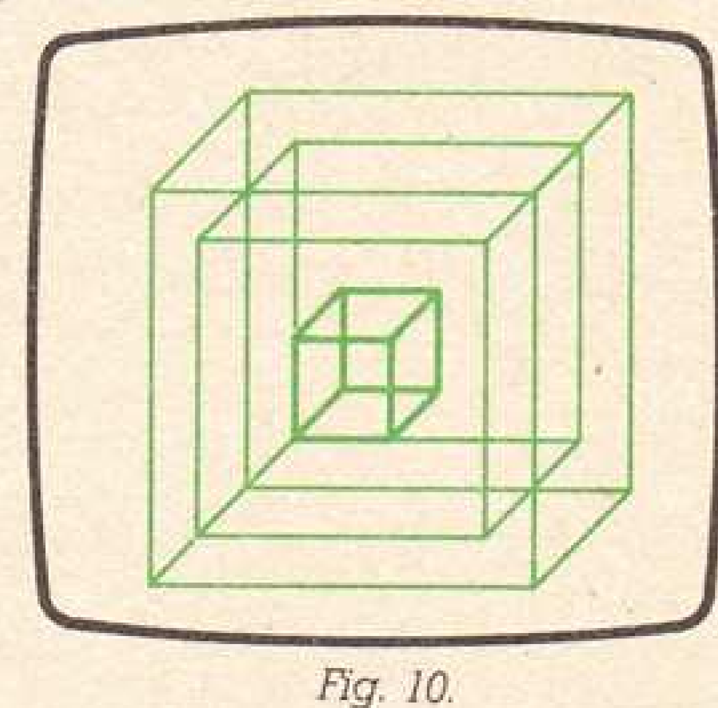


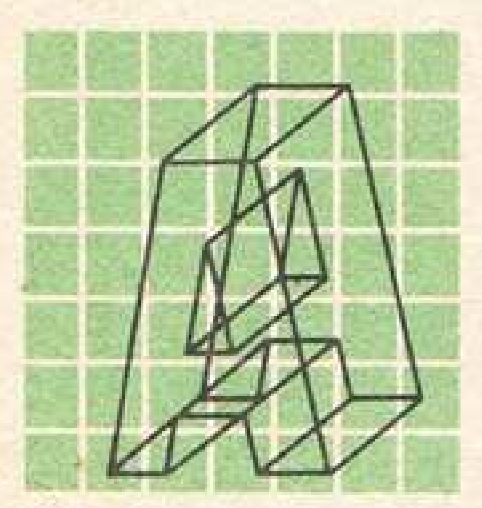
Fig. 10.

5:

- PONF: INCORRECTA. La orden es PONFONDO o bien PONCF.
- VANZA PONCL 40: INCORRECTA. A la orden AVANZA le falta un valor y a PONCL no se le puede asignar, en principio, un valor mayor de 15.
- PONX 600: CORRECTA.
- RELLENA 3: INCORRECTA. La orden RELLENA no admite ningún valor.
- AV AZAR 300: CORRECTA.
- PONFONDO 4 PONCL 3: CORRECTA.

Hay ordenadores en los que cada punto puede ser de un color. En otros esto no es posible.

Movimiento circular



NTES de empezar a leer esta sección es conveniente que repases algo las nociones básicas de trigonometría y de geometría. Si todavía no has estudiado estas ramas de la matemática en el colegio no importa. Cuando las estudies puedes darle un repaso a este tipo de movimiento.

Este movimiento no es muy usual en ninguno de los programas que se comercializan. Aun así, es necesario saber cómo realizarlo para, en cualquier momento, poder incorporar algún objeto que se mueva describiendo círculos en nuestros programas.

Si nos fijamos en la figura 1 podemos ver que alrededor de cada carácter de la pantalla hay otros ocho caracteres: dos verticales, dos horizontales y cuatro diagonales.

Podemos hacer que un asterisco los recorra uno a uno siguiendo la numeración que se ve en la figura 2.

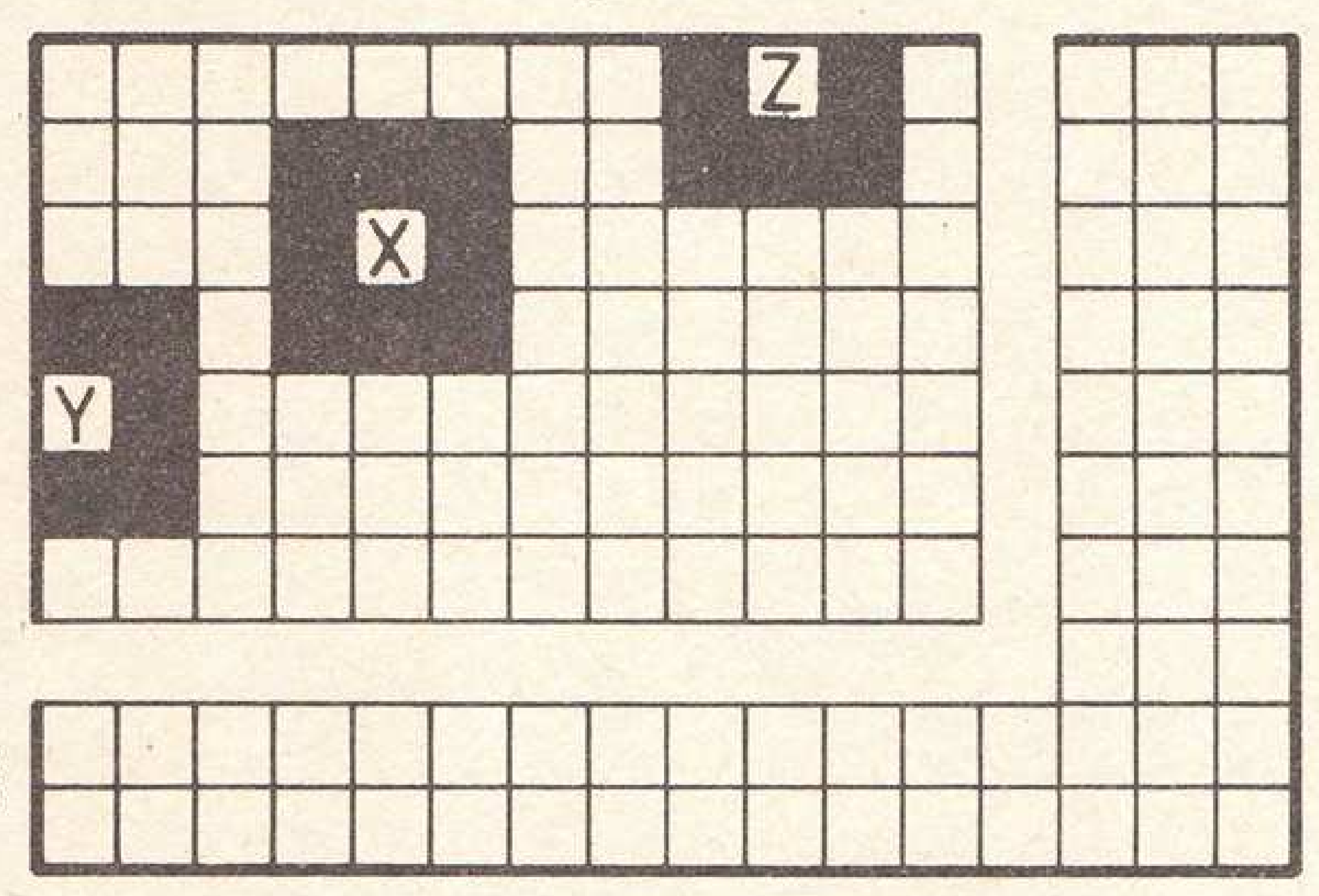


Fig. 1. Cada carácter "X" está rodeado de otros ocho a no ser que esté en un borde de la pantalla como "Y" y "Z".

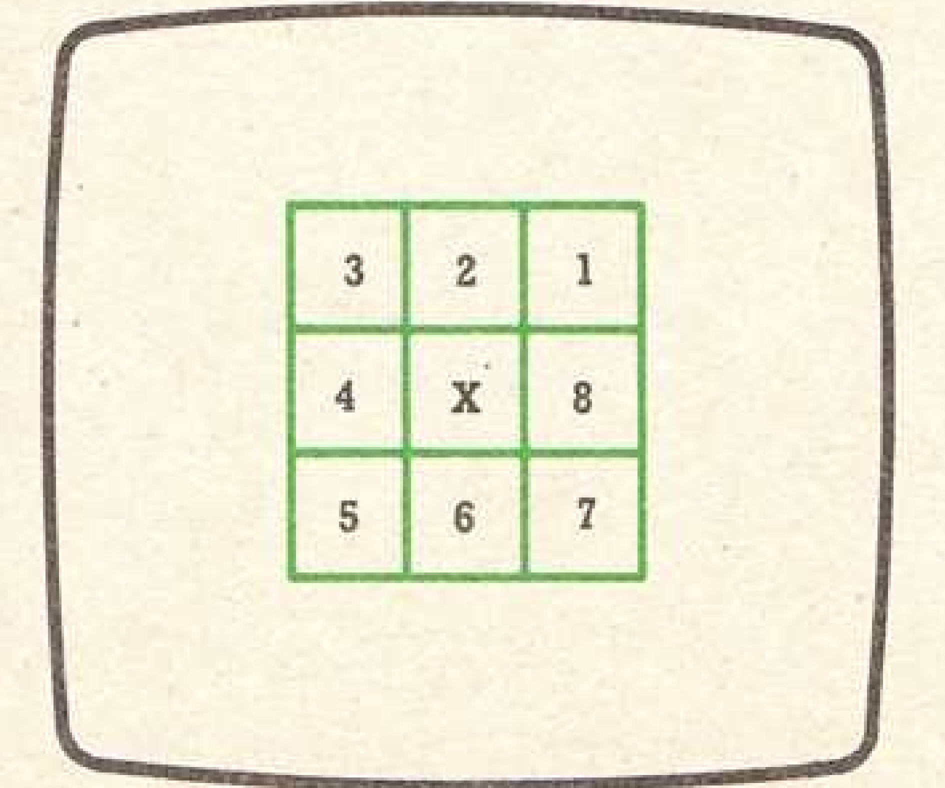


Fig. 2. El asterisco se moverá desde el cuadrado número 1 al número 8.

Para ver cómo quedaría este movimiento introduce y ejecuta el programa 1.

```
10 REM ********************************************
20 REM * MOVIMIENTO CIRCULAR DE UN ASTERISCO (1) *
30 REM ********************************************
40 REM
50 CLS
60 LET C=10:LET F=10
70 FOR I=1 TO 20
80     LOCATE F,C+1:PRINT "*":GOSUB 210:LOCATE F,C+1:PRINT " "
90     LOCATE F+1,C+1:PRINT "*":GOSUB 210:LOCATE F+1,C+1:PRINT " "
100    LOCATE F+1,C:PRINT "*":GOSUB 210:LOCATE F+1,C:PRINT " "
110    LOCATE F+1,C-1:PRINT "*":GOSUB 210:LOCATE F+1,C-1:PRINT " "
120    LOCATE F,C-1:PRINT "*":GOSUB 210:LOCATE F,C-1:PRINT " "
130    LOCATE F-1,C-1:PRINT "*":GOSUB 210:LOCATE F-1,C-1:PRINT " "
140    LOCATE F-1,C:PRINT "*":GOSUB 210:LOCATE F-1,C:PRINT " "
150    LOCATE F-1,C+1:PRINT "*":GOSUB 210:LOCATE F-1,C+1:PRINT " "
160 NEXT I
170 END
180 REM
190 REM *** RETARDO ***
200 REM
210 FOR J=1 TO 100:NEXT J:RETURN
```

Este programa funciona perfectamente y sin cambios en el IBM y Amstrad. Para el resto de los micros se tendrán que hacer las siguientes variaciones:

COMMODORE

Línea 50 PRINT "<SHIFT-HOME>"

En todas las líneas donde aparezca la sentencia 'LOCATE' hay que hacer un GOSUB a la línea 9900. Si, por ejemplo, tenemos la línea:

Línea 80 LOCATE F,C+1:
PRINT "*":GOSUB 210:
LOCATE F,C+1:PRINT " "

nosotros tendremos que sustituirla por:

Línea 80 X=C+1:Y=F:GOSUB 9900:PRINT "*":GOSUB 210:X=C+1:Y=F:GOSUB 9900:PRINT " "

También es necesario unir el programa 1 con la rutina 'LOCATE PARA COMMODORE' que se dio en el tomo 1.

MSX

Para este microordenador es necesario cambiar el orden de los argumentos de todas las sentencias 'LOCATE'. Si en el listado nos aparece:

LOCATE F,C+1

nosotros tendremos que poner:

LOCATE C+1,F

SPECTRUM

Hay que sustituir todas las sentencias LOCATE por PRINT AT de la siguiente manera:

LOCATE F,C+1

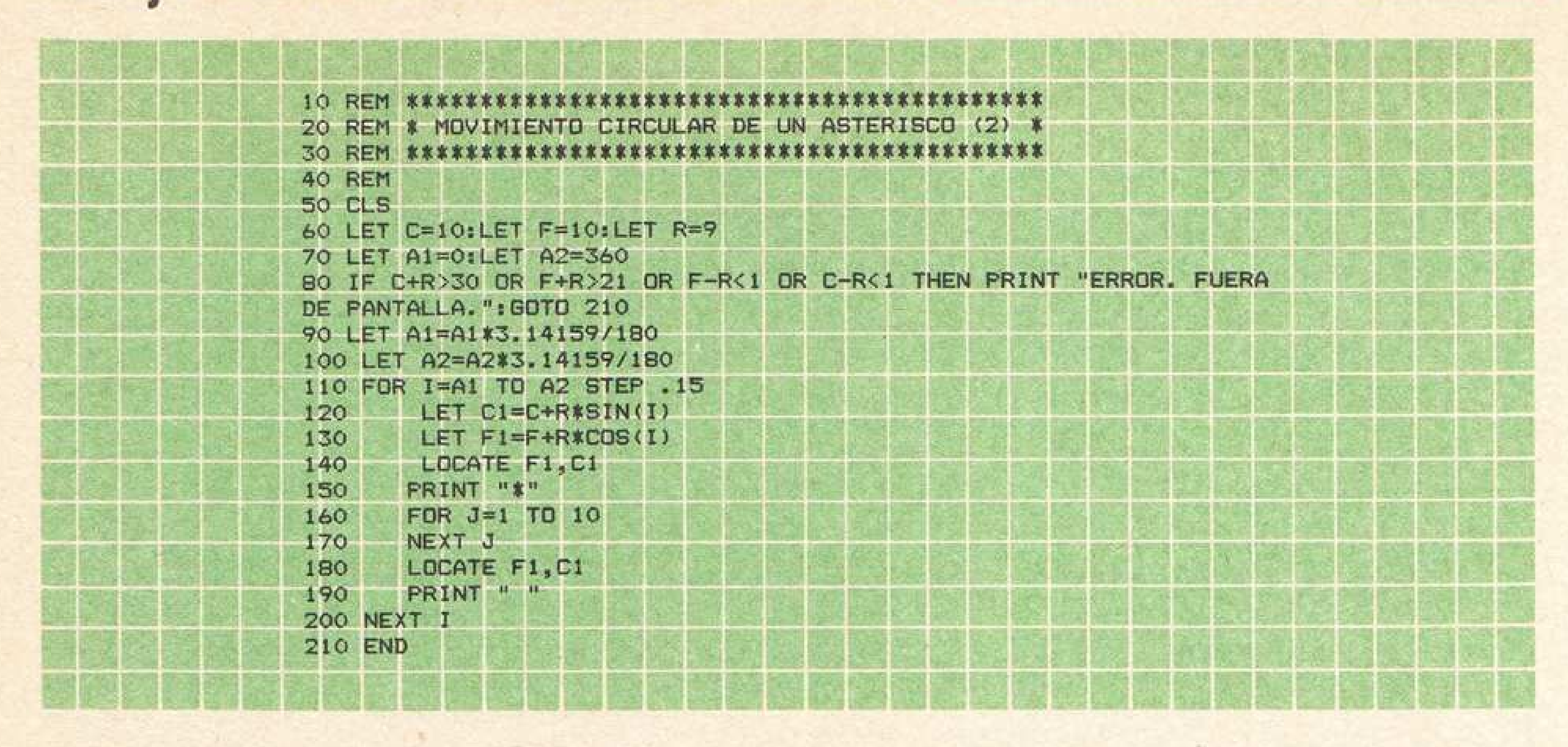
se convertiría en:

PRINT AT F,C+1;

sin olvidar un punto y coma (;) al final del segundo operando.

Como podéis ver, el asterisco (*) no se mueve alrededor de un círculo, sino de un cuadrado, pero el programa puede ser una buena aproximación a lo que estamos necesitando. Por otra parte, este programa sólo sirve para un caso muy especial, ya que variarlo supondría una gran cantidad de cambios.

Si quisiésemos un programa que nos moviese cualquier carácter por la pantalla con trayectoria circular, de cualquier radio, tendríamos que hacer un programa de uso general. A este programa habría que decirle el radio del círculo, el centro de dicho círculo y la longitud del arco que configura el movimiento. Este último valor nos dice si el movimiento es un círculo (y se cierra sobre sí mismo) o bien si es un trozo de un círculo (un arco de circunferencia).



```
10 REM ****************************************
20 REM * MOVIMIENTO CIRCULAR DE UN ASTERISCO (2) *
30 REM ****************************************
40 REM
50 CLS
60 LET C=10:LET F=10:LET R=9
70 LET A1=0:LET A2=360
80 IF C+R>30 OR F+R>21 OR F-R<1 OR C-R<1 THEN PRINT "ERROR. FUERA
DE PANTALLA.":GOTO 210
90 LET A1=A1*3.14159/180
100 LET A2=A2*3.14159/180
110 FOR I=A1 TO A2 STEP .15
120      LET C1=C+R*SIN(I)
130      LET F1=F+R*COS(I)
140      LOCATE F1,C1
150     PRINT "*"
160     FOR J=1 TO 10
170     NEXT J
180     LOCATE F1,C1
190     PRINT " "
200 NEXT I
210 END
```

Las modificaciones que hay que hacer para Commodore, MSX y Spectrum son las siguientes:

COMMODORE

Línea 50 PRINT "<SHIFT-HOME>"

También hay que sustituir todas las sentencias LOCATE por GOSUB 9900 tal y como se ha dicho un poco más arriba.

MSX

Es necesario cambiar de orden los argumentos de todas las sentencias LOCATE tal y cómo se ha visto un poco más arriba.

SPECTRUM

Hay que cambiar todas las sentencias LOCATE por PRINT AT tal y como se ha visto en el programa 1.

El funcionamiento del programa es el siguiente:

En la línea 10 se asigna a C y a F el valor de 10. Esta es la coordenada de pantalla que da el centro de la circunferencia sobre la que se realizará el movimiento. En esta línea también se define el radio (R) con un valor de 9.

La definición del ángulo de inicio y fin de movimiento se realiza en la línea 20. El ángulo ha de ir dado en grados, aunque más tarde se pasa a radianes.

En la línea 80 se comprueba si el círculo está dentro de los límites de la pantalla. En el caso de no ser así, se imprime un mensaje de error. El movimiento escapa de la pantalla cuando:

1. $C + R >$ Número máximo de caracteres por línea. Se sale por la derecha de la pantalla.
2. $C - R < 0$. Se sale por la izquierda de la pantalla.
3. $F + R >$ Número máximo de líneas en la pantalla. Se sale por debajo de la pantalla.
4. $F - R < 0$. Se sale por encima de la pantalla.

Donde:

C = Coordenada en X del centro de la circunferencia.

F = Coordenada en Y del centro de la circunferencia.

R = Radio de la circunferencia.

Las líneas 90 y 100 pasan los ángulos A1 y A2 que están en grados a radianes. Como todos sabemos, 180 grados es igual a PI radianes. Conociendo esto, para pasar cualquier grado a radianes sólo hay que hacer una regla de tres.

Si 180 grados → son PI radianes
entonces A1 grados → serán X radianes

Poniéndolo en forma de quebrado nos quedaría que:

$$\text{Radianes} = \frac{A1 * PI}{180}$$

En la línea siguiente (la 110) empieza un bucle que va desde el ángulo A1 hasta el A2.

Este bucle va dando la dirección, en grados, del movimiento. El incremento (STEP) está fijado en 0,15, pero se puede variar para conseguir que el movimiento sea más o menos definido y rápido.

Las líneas 120 y 130 calculan la columna y la fila donde tendremos que poner el asterisco (*). Para calcular la coordenada de la pantalla se suma a la posición central (F o C) el valor del seno (o del coseno) del ángulo actual multiplicado por el valor del radio. Para entender mejor esto fíjate en la figura 3.

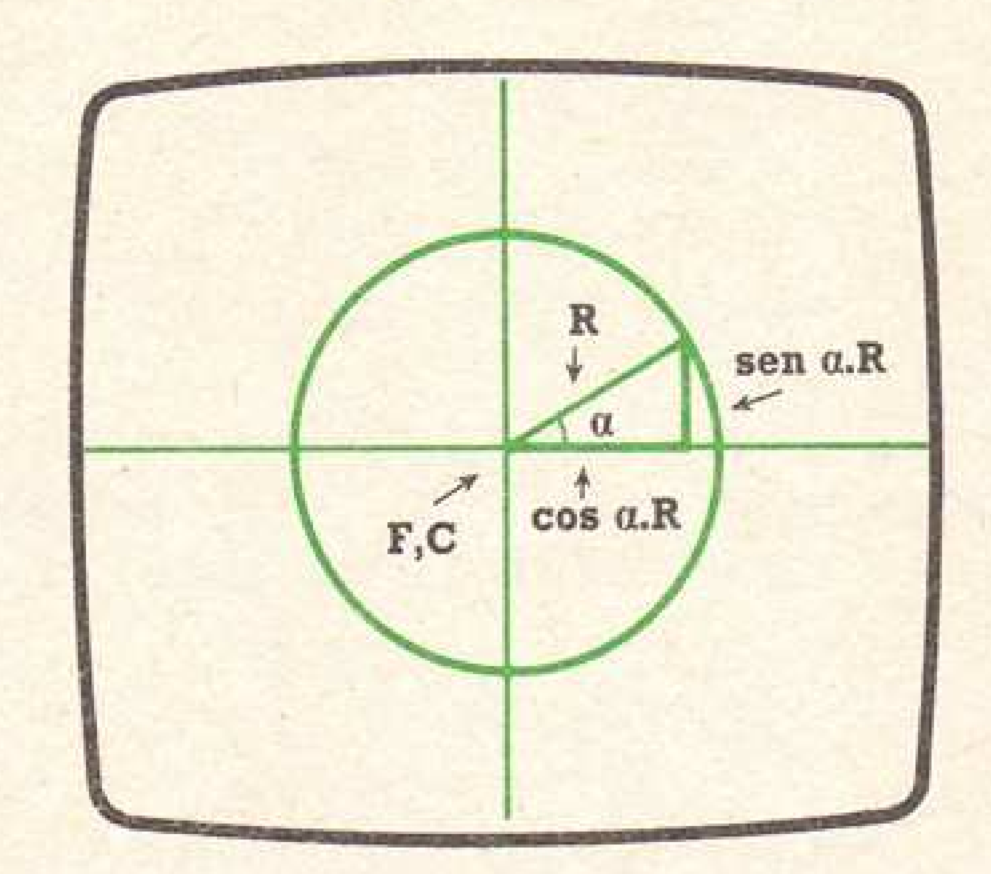


Fig. 3. Porque F1 = F + cos α.R y C1 = C + sen α.R

La línea 140 coloca el cursor en la columna y en la fila que acabamos de hallar y la línea 150 imprime un asterisco (*).

Como el asterisco tiene que permanecer un rato visible en la pantalla, en las líneas 160 y 170 se realiza un bucle vacío que actúa como un retardo de tiempo.

En las líneas 180 y 190 se borra el asterisco y en la siguiente línea (200) se continúa con el siguiente valor del ángulo.

Si se quiere que el asterisco no desaparezca al final del programa, sino que se quede en su última posición, habría que insertar la siguiente línea:

Línea 205 LOCATE F1,C1:PRINT "*"

Los usuarios del Spectrum, MSX y Commodore tendrán que remitirse, antes de introducir esta línea, a las modificaciones que se dieron anteriormente.

Para utilizar el programa sólo hay que decirle los siguientes datos:

R = Radio de la circunferencia o del arco.
A1 = Angulo inicial del movimiento.
A2 = Angulo final del movimiento.
F = Coordenada en Y del centro.
C = Coordenada en X del centro.

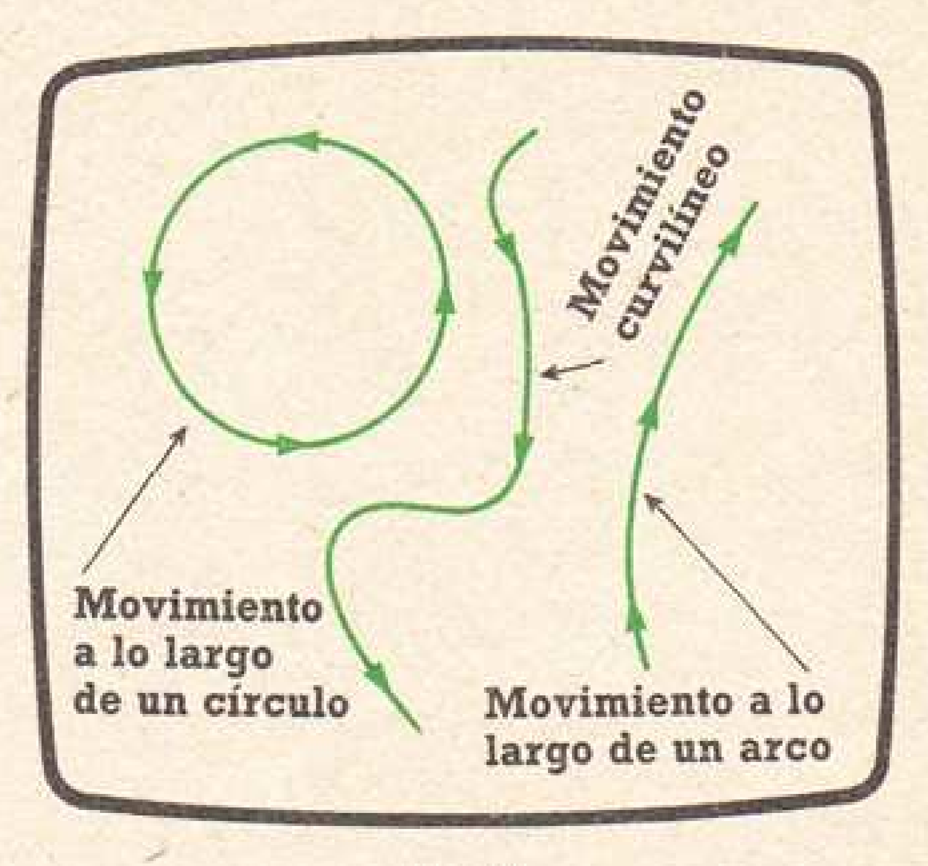


Fig. 4.

De esta manera, si decimos que A1 = 0 y A2 = 360, el asterisco realizará un círculo completo. Si A2 = 720, realizará dos círculos seguidos. Si A1 = 180 y A2 = 250, el movimiento seguirá la trayectoria de un arco de 70 grados. Prueba con diferentes valores y fíjate en la figura 4 para entender mejor todo esto.

Movimiento acelerado

El último movimiento que nos queda por ver es el movimiento acelerado (o decelerado). Este movimiento se puede dar tanto en el vertical como en el diagonal, horizontal y circular. Su realización es muy sencilla en comparación con los resultados que ofrece.

Como ejemplo tenemos el programa 3. En él veremos una O que se mueve de izquierda a derecha de la pantalla cada vez más deprisa.

```
10 REM ****************************************
20 REM * MOVIMIENTO RECTILINEO DE IZQUIERDA A *
30 REM * DERECHA UNIFORMEMENTE ACELERADO.     *
40 REM ****************************************
50 REM
60 CLS
70 LET Y=10
80 LET AC=114
90 LET DA=-3
100 FOR X=2 TO 39
110     LOCATE Y,X
120     PRINT " *"
130     FOR T=1 TO AC
140     NEXT T
150     LET AC=AC+DA
160 NEXT X
170 END
```

Las variaciones que es necesario realizar, para que este programa funcione en el MSX, Spectrum y Commodore, son las que se dan a continuación:

COMMODORE

Línea 60 PRINT "<SHIFT-HOME>"
Línea 110 GOSUB 9900

Es necesario unir el programa con la rutina LOCATE PARA COMMODORE que se dio en la sección de trucos de programación en el tomo primero.

SPECTRUM

Línea 100 FOR X=1 TO 30
Línea 110 PRINT AT Y,X;

MSX

Línea 110 LOCATE X,Y

Como puedes apreciar en el programa, el asterisco se mueve por la pantalla cada vez más deprisa. Esto se logra variando la longitud del bucle de retardo que se encuentra en las líneas 130 y 140.

Al principio del programa la variable AC (ACeleración) tiene valor 114. En la primera vuelta del bucle principal, después de imprimir el asterisco, se espera durante 114 pasos del segundo bucle. En la línea 150 se resta de la variable AC el valor de DC (Diferencial de Aceleración) y se repiten de nuevo todas las líneas anteriores.

Con esto se consigue que AC tenga cada vez un valor menor, por lo que el bucle de las líneas 130 y 140 cada vez es más rápido y, por tanto, el asterisco cada vez va más deprisa.

Para conseguir que el movimiento sea decelerado, en vez de acelerado, lo único que hay que hacer es cambiar las siguientes líneas:

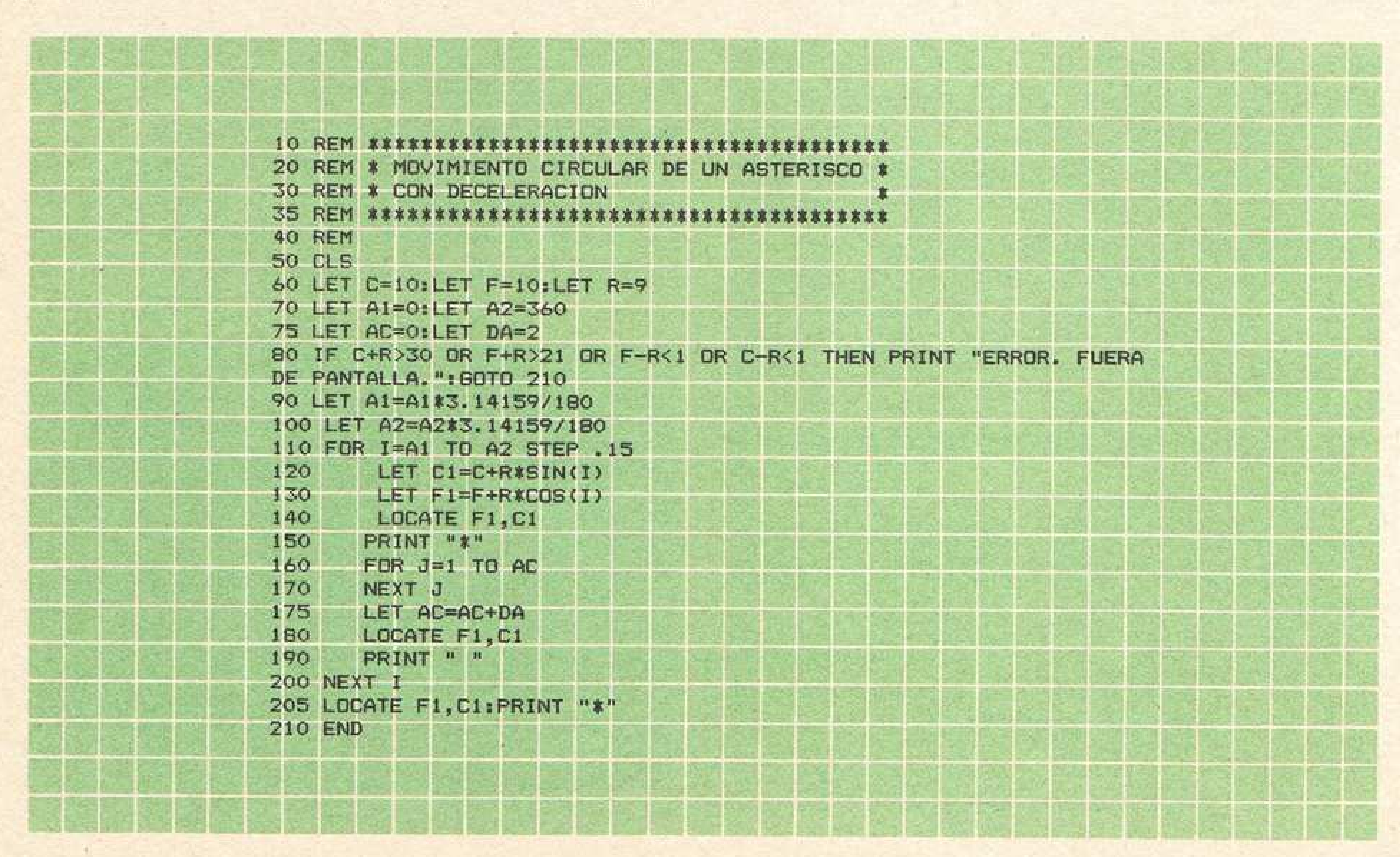
Línea 80 LET AC=0
Línea 90 LET DA=3

Con esto el valor de AC parte desde cero y va aumentando en cada vuelta del bucle principal.

Si en un momento dado se quiere que el movimiento sea uniforme (que no acelera ni decelera con el tiempo) habrá que darle a las variables AC y DA el valor cero.

Línea 80 LET AC=0
Línea 90 LET DA=0

Este pequeño truco sirve igual para cualquier tipo de movimiento de los que hemos visto. A continuación aparece el programa del movimiento circular con las variaciones necesarias para que sea uniformemente decelerado.



```
10 REM ***************************************
20 REM * MOVIMIENTO CIRCULAR DE UN ASTERISCO *
30 REM * CON DECELERACION                    *
35 REM ***************************************
40 REM
50 CLS
60 LET C=10:LET F=10:LET R=9
70 LET A1=0:LET A2=360
75 LET AC=0:LET DA=2
80 IF C+R>30 OR F+R>21 OR F-R<1 OR C-R<1 THEN PRINT "ERROR. FUERA
DE PANTALLA.":GOTO 210
90 LET A1=A1*3.14159/180
100 LET A2=A2*3.14159/180
110 FOR I=A1 TO A2 STEP .15
120      LET C1=C+R*SIN(I)
130      LET F1=F+R*COS(I)
140      LOCATE F1,C1
150     PRINT "*"
160     FOR J=1 TO AC
170     NEXT J
175     LET AC=AC+DA
180     LOCATE F1,C1
190     PRINT " "
200 NEXT I
205 LOCATE F1,C1:PRINT "*"
210 END
```

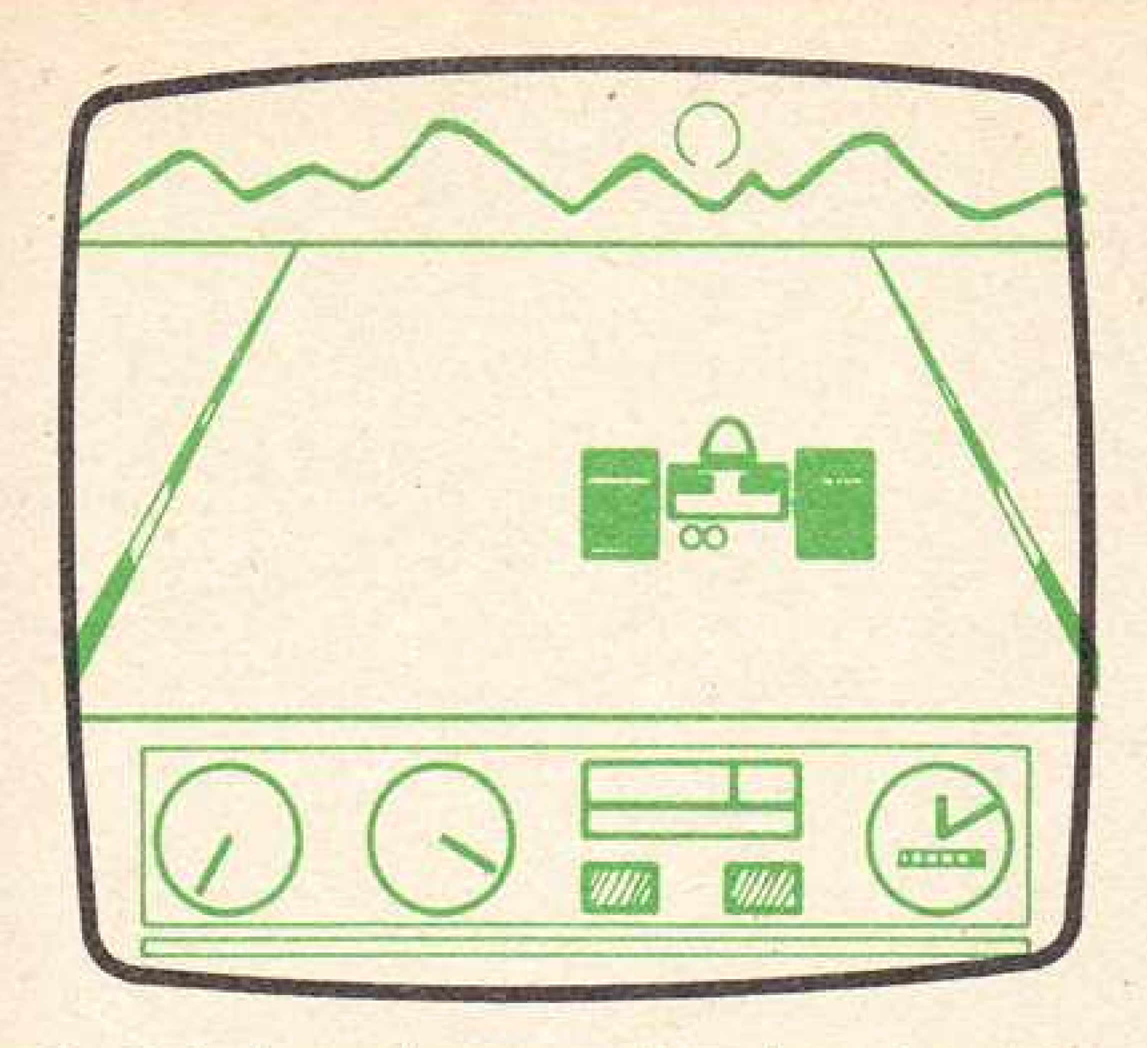


Fig. 5. En los juegos de carreras de coches y de motos es necesario incluir rutinas de movimiento uniformemente acelerado para que el juego pueda parecerse a la realidad.

Choques y rebotes

Es lógico pensar que en cualquier juego que queramos realizar en algún momento de su desarrollo se producirá un choque de algún cuerpo con otro o contra los límites de la pantalla.

En todos los choques que se produzcan se da la circunstancia de que el movimiento cambia de sentido. Esto es, si un carácter se mueve de derecha a izquierda, y llega al extremo izquierdo de la pantalla, el movimiento se invertirá y será de izquierda a derecha. Si un movimiento diagonal positivo-positivo choca contra una pared se volverá negativo-positivo o positivo-negativo, según de qué pared se trate.

Veamos un programa de cada uno de estos tipos.

```
10 REM ********************************************
20 REM * CHOQUE CONTRA UNA PARED Y CAMBIO DEL SEN *
30 REM * TIDO DEL MOVIMIENTO                      *
40 REM ********************************************
50 REM
60 CLS
70 REM
80 REM *** INICIALIZACION DE VARIABLES ***
90 REM
100 LET Y=10
110 LET C1=29
120 LET C2=2
130 LET C3=C1
140 LET ST=-1
150 REM
160 REM *** DIBUJO DE LAS PAREDES LATERALES ***
170 REM
180 FOR I=1 TO 20
190     LOCATE I,1
200     PRINT "#"
210     LOCATE I,30
220     PRINT "#"
230 NEXT I
240 REM
250 REM *** MOVIMIENTO EN LA DIRECCION ST ***
260 REM
270 REM
280 FOR I=C1 TO C2 STEP ST
290 IF I=C1 THEN GOTO 320
300     LOCATE Y,I-ST
310     PRINT " "
320     LOCATE Y,I
330     PRINT "."
340     FOR J=1 TO 100
350     NEXT J
360 NEXT I
370 REM
380 REM *** CAMBIO DE SENTIDO ***
390 LET C1=C2
400 LET C2=C3
410 LET C3=C1
420 LET ST=-ST
430 GOTO 280
```

Como siempre, las variaciones necesarias para que el programa 5 funcione en ordenadores distintos del Amstrad y del IBM son:

COMMODORE

Línea 60 PRINT "<SHIFT-HOME>"
Línea 190 X=1:Y=I:GOSUB 9900
Línea 210 X=30:Y=I:GOSUB 9900
Línea 300 X=I-ST:GOSUB 9900
Línea 320 X=I:GOSUB 9900

Por supuesto, y como siempre, hay que unir el programa con la rutina LOCATE PARA COMMODORE del tomo número uno.

MSX

Línea 190 LOCATE 1,I
Línea 210 LOCATE 30,I
Línea 300 LOCATE I-ST,Y
Línea 320 LOCATE I,Y

SPECTRUM

Línea 190 PRINT AT I,1;
Línea 210 PRINT AT I,30;
Línea 300 PRINT AT Y,I-ST;
Línea 320 PRINT AT Y,I;

Las correcciones que hay que hacerle al programa 6 son las siguientes:

COMMODORE

Línea 70 PRINT "<SHIFT-HOME>"
Línea 120 X=1:Y=I:GOSUB 9900

```
10 REM ***********************************************
20 REM * MOVIMIENTO DIAGONAL CON REBOTE EN LAS PAREDES *
30 REM ***********************************************
40 REM
50 LET X=10:LET Y=10
60 LET X1=1:LET Y1=1
70 CLS
80 FOR I=1 TO 30
90    PRINT "#";
100 NEXT I
110 FOR I=2 TO 19
120    LOCATE I,1
130    PRINT "#"
140    LOCATE I,30
150    PRINT "#"
160 NEXT I
170 FOR I=1 TO 30
180    PRINT "#";
190 NEXT I
200 REM
210 REM *** MOVIMIENTO ***
220 REM
230 LOCATE Y,X
240 PRINT "O"
250 FOR J=1 TO 100
260 NEXT J
270 LOCATE Y,X
280 PRINT " "
290 LET X=X+X1:LET Y=Y+Y1
300 IF X>28 OR X<3 THEN LET X1=-X1
310 IF Y>18 OR Y<3 THEN LET Y1=-Y1
320 GOTO 230
```

Programa 6.

Línea 140 X=30:Y=I:GOSUB 9900
Línea 230 GOSUB 9900
Línea 270 GOSUB 9900

Hay que unir este programa con la rutina LOCATE PARA COMMODORE.

MSX

Línea 120 LOCATE 1,I
Línea 140 LOCATE 30,I
Línea 230 LOCATE X,Y
Línea 270 LOCATE X,Y

SPECTRUM

Línea 120 PRINT AT I,1;
Línea 140 PRINT AT I,30;
Línea 230 PRINT AT Y,X;
Línea 270 PRINT AT Y,X;

Con el programa 5 podemos ver cómo un puntito (la famosa estrella fugaz) se mueve por la pantalla de izquierda a derecha. Cuando llega a la barrera de almohadillas (#) invierte el sentido del movimiento y se mueve de derecha a izquierda. Cuando encuentra otro muro de almohadillas vuelve a invertir el sentido del movimiento. Esto se va realizando una vez tras otra, indefinidamente.

Entre las líneas 180 y 220 imprimimos los dos muros (derecho e izquierdo) sobre los que chocará nuestra estrella.

A partir de la línea 280 y hasta el final del programa se realiza el movimiento. La clave del sentido de éste radica en la variable numérica ST. Esta variable puede tener dos valores, 1 y -1. Al colocarla como contador de incremento de la sentencia FORNEXT de la línea 280 hace que el bucle sea creciente (1) o decreciente (-1).

Los argumentos de la sentencia FORNEXT también van variando. Esto es necesario porque si el incremento del STEP es positivo entonces C1 ha de ser mayor que C2. Pero si el incremento es negativo entonces C2 ha de ser mayor que C1. Para solucionar esto se emplea una variable auxiliar (C3) gracias a la cual, cuando el punto llega a uno de los dos muros, C1 será igual a C2, y C2 igual a C1.

El resto del programa no reviste ninguna complicación, pues es igual a todo lo que hemos visto hasta ahora.

El programa 6 es más bonito y el resultado es más agradable y espectacular que el

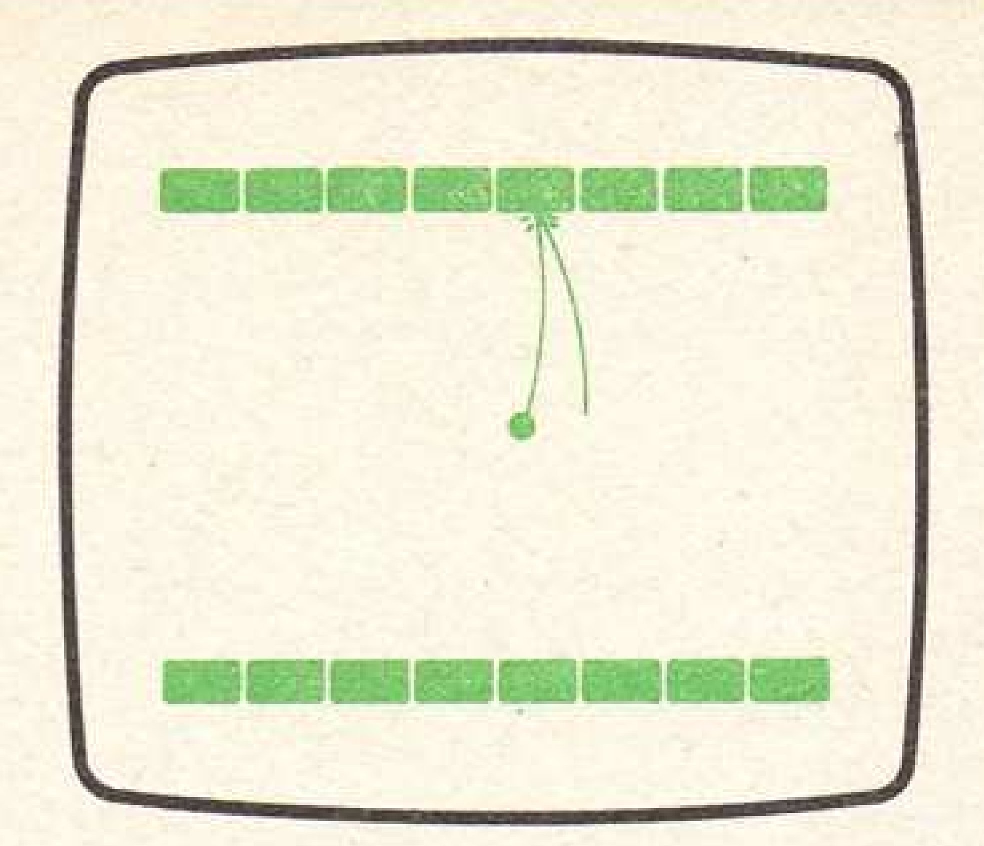


Fig. 6. Cada vez que el puntito choca contra una pared, el movimiento cambia de sentido.

anterior. En este programa la dificultad estaba en que, al ser un movimiento diagonal de 45 grados, hay cuatro posibles direcciones.

1. Positivo-positivo.
2. Positivo-negativo.
3. Negativo-positivo.
4. Negativo-negativo.

En cualquier momento, y dependiendo del tipo de dirección que llevase la pelota (una letra O) al moverse, había que cambiar la dirección de ésta. Una solución podría haber sido el estar pendiente del tipo de dirección que hay en cada momento para así saber cuál es la que tendríamos que aplicar después de cada choque.

Como se puede ver en el listado del programa, la solución es mucho más sencilla.

Todo consiste en comprobar si la pelota ha llegado a alguna de las paredes. Como tenemos cuatro paredes, se nos presentan cuatro casos:

1. Choque con la pared izquierda.
2. Choque con la pared derecha.
3. Choque con la pared superior.
4. Choque con la pared inferior.

que corresponden a unos ciertos valores de las variables X e Y, que son las que almacenan la posición de la bola en cualquier momento. Dichos valores, ordenados según la posición de la pelota, son:

1. X = 2 → Pared izquierda.
2. X = 29 → Pared derecha.
3. Y = 2 → Pared superior.
4. Y = 19 → Pared inferior.

Por otro lado, se puede apreciar que cada vez que la bola llega a una de las cuatro paredes, el movimiento cambia de sentido, sólo en la componente a la que pertenece dicha pared. Así, si la bola llega a una de las paredes laterales sólo variará la componente en X de su movimiento.

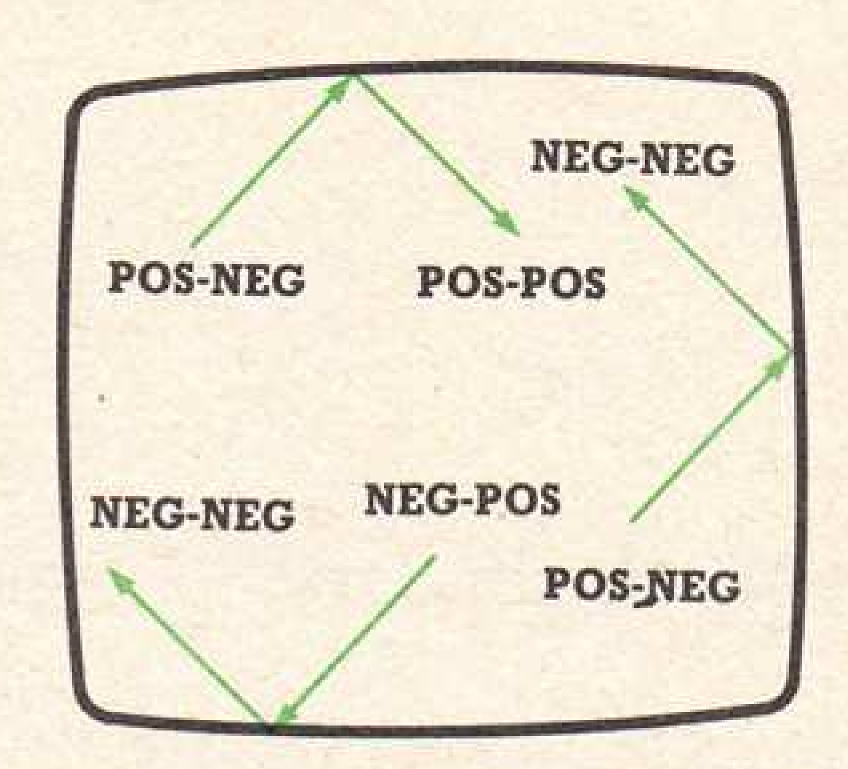


Fig. 7. Como se puede apreciar en el dibujo, después de cada choque sólo cambia una de las componentes.

Según todo lo visto, lo único que hay que hacer cada vez que la pelota choca contra una pared es cambiar el sentido del movimiento en la dirección de la pared.

Todo lo que hemos visto sobre el programa 6 lo realizan sólo dos líneas. Estas son:

Línea 300 IF X>28 OR X<3 THEN X1=-X1
Línea 310 IF Y>18 OR X<3 THEN Y1<-Y1

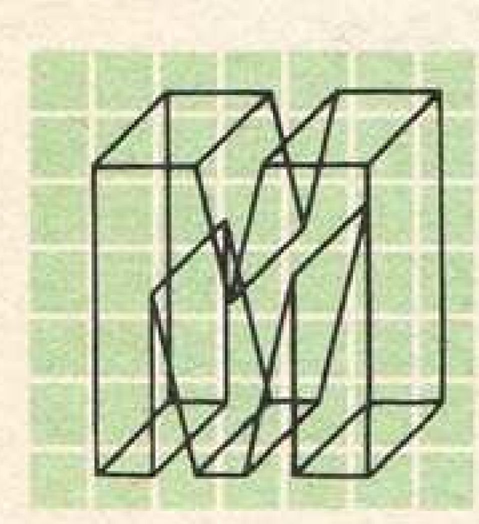
Lo que hacen es comprobar si la pelota ha chocado con algunas de las paredes y, en caso afirmativo, variar la dirección (X1 o Y1, según el caso).

Nota importante

Lo más importante, a la hora de programar, es tener todos los conceptos muy claros. Esto se consigue de dos maneras: la primera, estudiando y viendo la parte teórica de los problemas; y la segunda, y más importante, practicando y realizando programas cada vez más complicados que nos ayuden a asentar todos los conocimientos teóricos.

Por todo ello, te recomiendo que no te limites sólo a leer lo que aquí se da y a introducir los programas que aquí proponemos, sino que tú mismo te plantees problemas a resolver y los resuelvas. Esto no solamente te hará entender mucho mejor lo que hayas leído, sino que además te capacitará para poder entender conceptos más avanzados y complicados que iremos viendo en tomos sucesivos.

Introducción de claves de seguridad



UCHAS veces es necesario que un programa no pueda ser utilizado nada más que por un cierto número de personas. Estas personas deben de conocer una clave de acceso que tendrán que introducir al principio de la ejecución del programa para que éste empiece a funcionar. Incluso se da el caso de que un programa muy importante y complejo necesita de las claves de más de una persona a la vez para poder funcionar. Esto sirve para que una persona no autorizada no pueda utilizarlo.

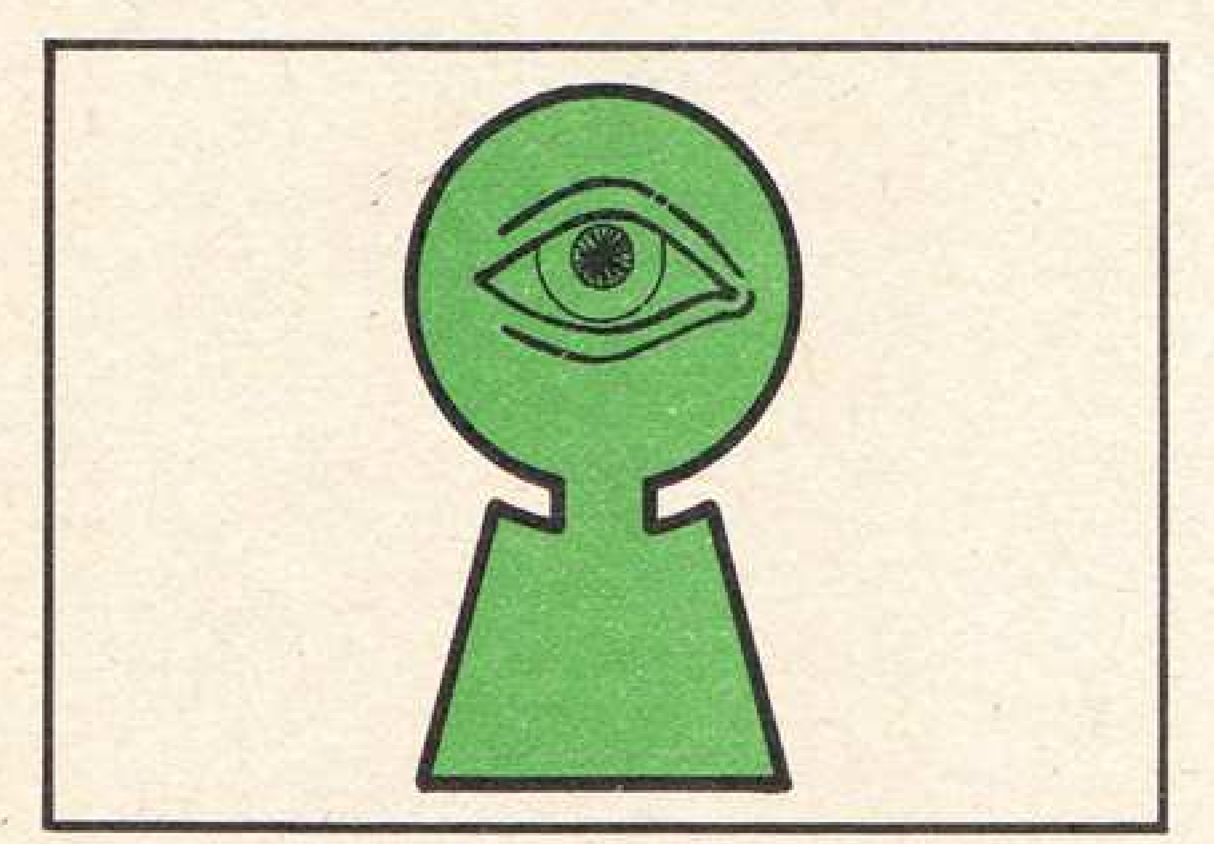


Fig. 1. La introducción de claves de seguridad en los programas impide que miradas indiscretas hagan uso de éstos.

Este tipo de protecciones se dan mucho en todas las empresas de software, donde cada programador tiene asignados una serie de programas de los que sólo él es reponsable. Debido a la gran cantidad de información confidencial que hay en estas empresas o en organismos públicos, como puede ser el ejército, es necesario restringir la entrada a ciertos datos y programas a sólo unos cuantos.

Los programas y trucos que veremos a continuación no son de una gran complicación, pero nos permitirán hacer de nuestros programas algo más íntimo y más nuestro. Los programas que aquí se van a proponer no son sino una parte de los que ya existen y de los que tú mismo te puedes inventar y realizar con un poco de práctica e imaginación.

Antes de empezar tenemos que diferenciar entre las protecciones que se realizan para salvaguardar datos y las que sirven para los programas.

Los datos pueden ser almacenados de una forma codificada, de tal manera que para poder leerlos hace falta decodificarlos. En este caso la única dificultad estriba en buscar un método de codificación que sea lo más sencillo posible (para que no se lleve mucho tiempo de ejecución). Además es necesario que la decodificación, aparte de rápida, precise de algún dato externo con el que se llevará a efecto. Este dato (o datos) externo debe ser introducido por el usuario, como una clave, al principio del programa.

Por otro lado están los programas. En ellos la dificultad se encuentra en que es con-

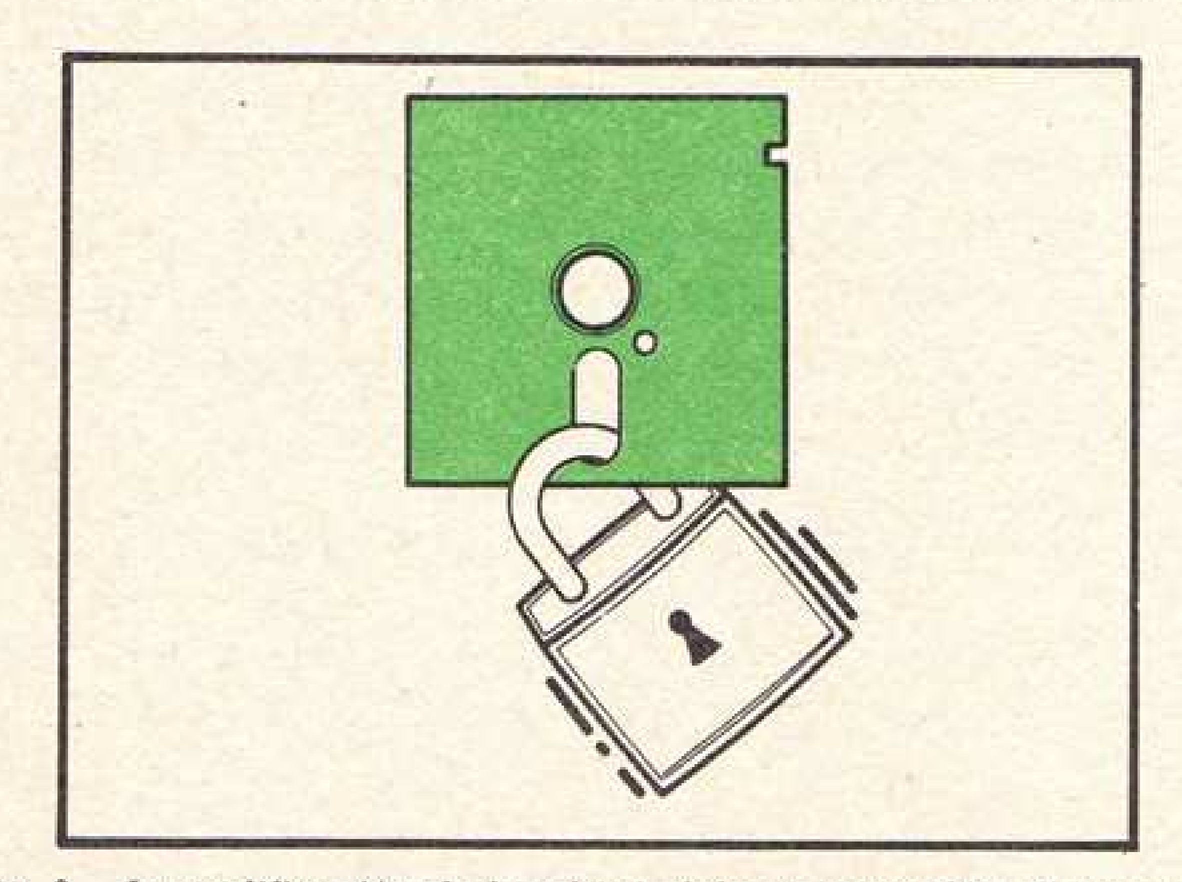


Fig. 2. La codificación de los datos debe ir precedida de la protección y cierre de los programas que decodifican dichos datos.

veniente que se autoejecuten cuando se cargan en memoria. En caso contrario, cualquier persona sería capaz de listar o mirar el programa buscando la parte de este donde se hace la entrada del código de seguridad, neutralizándolo o aprendiendo cuál es éste, para cambiarlo o utilizarlo tal y como está.

Nuestro primer programa será una rutina que nos pedirá un código de entrada antes de empezar con la ejecución del programa.

```
10 REM **********************************************************
11 REM * <<< SUBRUTINA DE ENTRADA DE CODIGO DE SEGURIDAD >>> *
12 REM *                                                        *
13 REM * VALIDA PARA MSX, AMSTRAD, IBM, COMMODORE Y SPECTRUM *
14 REM *                                                        *
15 REM **********************************************************
16 REM *                                                        *
17 REM * VARIABLES QUE HAY QUE PASARLE A LA RUTINA             *
18 REM * ----------------------------------------              *
19 REM *                                                        *
20 REM * NINGUNA                                                *
21 REM *                                                        *
22 REM * EL PROGRAMA NO DEVUELVE NINGUN VALOR                   *
23 REM *                                                        *
24 REM * VARIABLES USADAS INTERNAMENTE                          *
25 REM * ----------------------------                           *
26 REM *                                                        *
27 REM * A$ = ALMACENA EL PASSWORD QUE SE INTRODUCE            *
28 REM * B$ = ALMACENA EL PASSWORD A INTRODUCIR                *
29 REM * C$ = RECOGE CARACTERES DEL TECLADO                    *
30 REM * I  = CONTADOR DE BUCLE                                 *
31 REM *                                                        *
32 REM **********************************************************
33 REM
40 LET A$=""
41 LET B$="Password"
42 CLS
43 PRINT "POR FAVOR. INTRODUZCA EL CODIGO DE ACCESO"
44 PRINT:PRINT "---> _";CHR$(29);
45 FOR I=1 TO LEN(B$)
46    LET C$=INKEY$:IF C$="" THEN GOTO 46
47    PRINT CHR$(65+RND*25);"_";CHR$(29);
48    LET A$=A$+C$
49 NEXT I
50 CLS
51 PRINT "CODIGO DE ACCESO ERRONEO"
52 FOR I=1 TO 200:NEXT I
53 IF A$<>B$ THEN GOTO 50
54 CLS
55 PRINT "CODIGO DE ACCESO CORRECTO"
56 PRINT
57 PRINT "PULSE 'C' PARA EMPEZAR LA EJECUCION DEL PROGRAMA"
58 LET C$=INKEY$:IF C$<>"C" THEN GOTO 58
59 CLS
```

Este programa se encuentra a partir de la línea n.° 10, justamente principio del programa. Esto se puede cambiar, por ejemplo, a la línea 4500, pero es necesario poner en la primera línea **GOSUB N.° de línea de inicio,** en nuestro caso **GOSUB 4500.**

Las variaciones que son necesarias realizar dependiendo del tipo de ordenador son las que aparecen a continuación:

COMMODORE

```
42 PRINT "<SHIFT-HOME>"
44 PRINT:PRINT "--->_";"<CURSOR
IZQUIERDA>";
46 GET C$:IF C$="" THEN GOTO 46
47 PRINT CHR$(65+RND(1)*25);" ";"<CURSOR
IZQUIERDA>";
50 PRINT "<SHIFT-HOME>"
54 PRINT "<SHIFT-HOME>"
58 GET C$:IF C$<>"C" THEN GOTO 58
59 PRINT "<SHIFT-HOME>"
```

MSX

```
47 PRINT CHR$(65+RND(1)*25);"_";CHR$(29);
```

SPECTRUM

```
44 PRINT:PRINT "--->_";CHR$(8);
47 PRINT CHR$(65+RND*25);"_";CHR$(8);
```

AMSTRAD

```
47 PRINT CHR$(65+RND(1)*25);"_":CHR$(29);
```

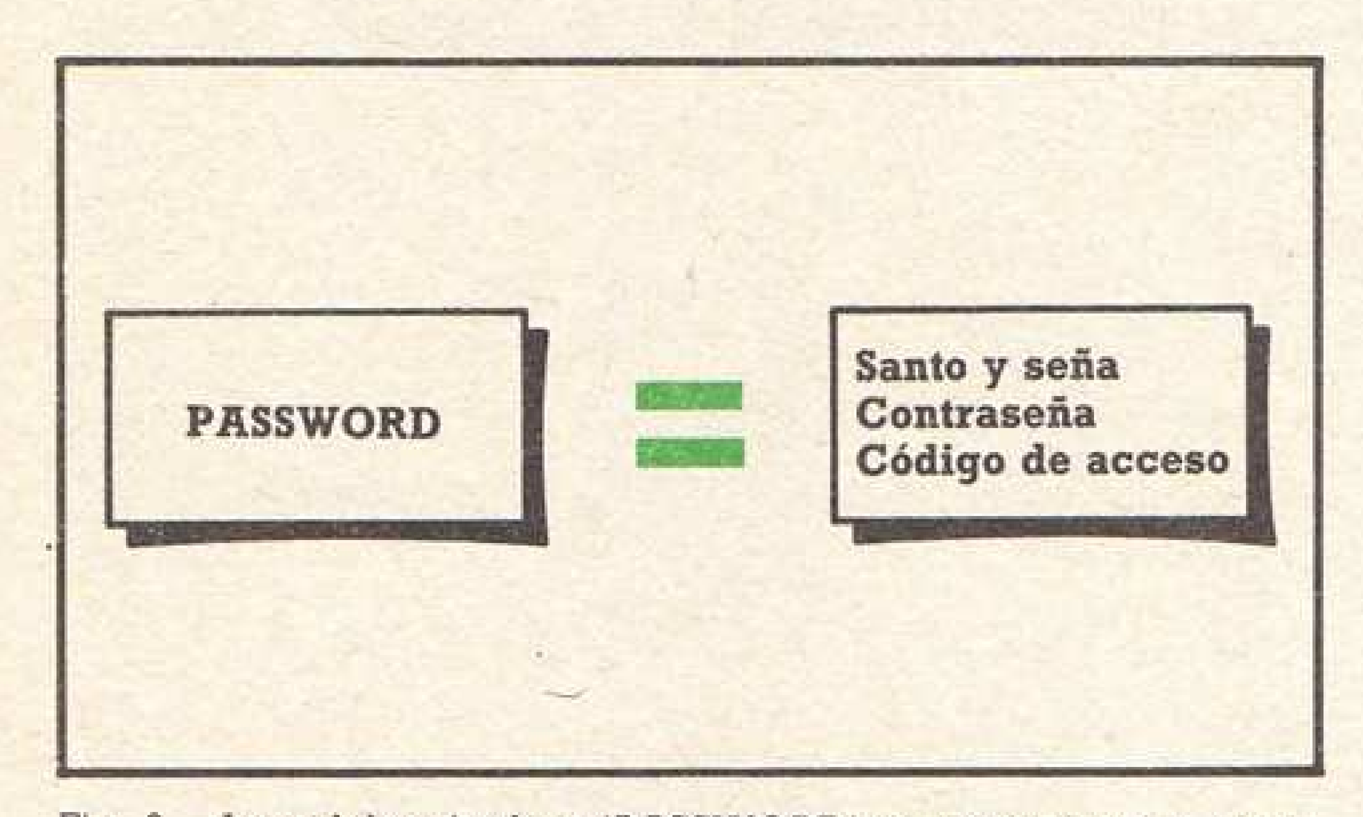


Fig. 3. La palabra inglesa 'PASSWORD' es equivalente a la española 'CONTRASEÑA'.

El funcionamiento del programa línea a línea es el siguiente:

Línea 40. Inicializa la variable A$ que será la encargada de almacenar el código de acceso que se introduce desde el teclado.

Línea 41. Asigna a la variable numérica B$ el string Password. Esto lo puedes cambiar por la contraseña que tú prefieras.

Línea 42. Borra la pantalla.

Línea 43. Imprime un mensaje en pantalla.

Línea 44. Imprime una flecha y el cursor. El hecho de poner al final de la línea CHR$(29) (o CHR(8) en el SPECTRUM) sirve para mover el cursor un lugar hacia la izquierda. Con esto se logra que éste se coloque en donde se encuentra el dibujo del cursor.

Línea 45. Empieza un bucle desde uno hasta la longitud de la clave de seguridad dentro del cual se pedirá, letra a letra, dicha clave.

Línea 46. Se mira si se ha pulsado alguna tecla. Si se pulsó, se almacena en C$. En caso contrario se vuelve a preguntar hasta que se pulse alguna.

Línea 47. Se imprime una letra comprendida de la A a la Y aleatoriamente usando la función RND. Esto sirve para que ninguna mirada indiscreta pueda aprender nuestra clave.

Línea 48. Se concatena la variable A$ con la última letra leída del teclado y que está almacenada en C$.

Línea 49. Aquí termina el bucle.

Línea 50. Se borra pantalla.

Línea 51. Se imprime el mensaje:

Código de acceso erróneo

Si la clave resultó ser correcta, borremos este mensaje. En caso contrario, le haremos que parpadee.

Línea 53. Se comprueba si la clave introducida es igual a la que se pedía. En caso contrario se vuelve a la línea 50. Como entre la línea 50 y la 53 no hay ninguna entrada de claves este bucle será, lo que se llama, un bucle infinito.

Línea 54. Se borra pantalla. Esto se realiza cuando la clave introducida coincide con la que se pedía. Gracias a esto se borra el mensaje que decía que el código de acceso era erróneo.

Línea 55. Se imprime un mensaje diciendo que la clave es correcta.

Línea 56. Dejamos una línea en blanco en la pantalla.

Línea 57. Aparece un mensaje que le dice al usuario que pulse la letra 'C' para empezar la ejecución del programa.

Línea 58. Se mira en el teclado hasta que se ha pulsado la letra 'C'.

Línea 59. Se borra la pantalla.

A partir de este punto es donde tiene que venir tu programa. En el caso de que esta rutina no la localices al principio del progra-

ma, y la llames mediante un **GOSUB N.° de línea,** tendrás que añadir a la rutina una última línea que ponga **RETURN.**

Como puedes apreciar, esta subrutina es muy sencilla, pero cumple perfectamente con el propósito encomendado.

Recuerda que la clave de acceso puede ser todo lo larga y complicada que tú quieras.

Ten en cuenta que dicha clave puede estar compuesta por una serie de caracteres que tú puedas introducir por el teclado.

Estos caracteres son los siguientes:

Los números del 0 al 9.
Las letras mayúsculas de la A a la Z.
Las letras minúsculas de la a a la z.
Los signos de puntuación como , . " ; : ' ' ^ etc.
Algunos de los caracteres especiales como ¡ ¿ # $ % / & * (,) - + < >, etc.

A continuación aparecen dos programas para la codificación y decodificación de un mensaje. Estos dos programas, unidos a los que tú realices, los hará más seguros e inaccesibles.

El primero de ellos (programa n.° 2) sirve para codificar una clave. Este programa no tiene que estar unido al que tú realices, sino que lo tienes que utilizar para crear una línea DATA en la que se guardará el mensaje codificado.

Mensaje	Mensaje codificado
HOLA	RYVK
CONTRASEÑA	HTSYWFXISF
12-12-1986	C07CD7CLKI

Fig. 4. Al codificar un mensaje, éste se vuelve incomprensible a primera vista.

```
100 REM ******************************************************
101 REM *  <<< CODIFICADOR DE MENSAJES POR EL METODO XOR >>>  *
102 REM *                                                     *
103 REM * VALIDO PARA AMSTRAD, IBM, MSX, COMMODORE Y SPECTRUM *
104 REM *                                                     *
105 REM ******************************************************
106 REM *                                                     *
107 REM * VARIABLES QUE HAY QUE PASARLE A LA RUTINA           *
108 REM * ------------------------------------------          *
109 REM *                                                     *
110 REM *  NINGUNA. EL MENSAJE SE INTRODUCE EN MEMORIA MEDIAN-*
111 REM * TE UNA SENTENCIA 'INPUT'                            *
112 REM *                                                     *
113 REM * EL RESULTADO APARECE EN PANTALLA                    *
114 REM *                                                     *
115 REM * VARIABLES USADAS INTERNAMENTE                       *
116 REM * ----------------------------                        *
117 REM *                                                     *
118 REM * A$ = ALMACENA EL MENSAJE INTRODUCIDO                *
119 REM * B$ = ALMACENA EL MENSAJE CODIFICADO                 *
120 REM * N  = NUMERO ALEATORIO                               *
121 REM * I  = CONTADOR DE BUCLE                              *
122 REM * N1 = VARIABLE AUXILIAR                              *
123 REM * N2 = VARIABLE AUXILIAR                              *
124 REM *                                                     *
125 REM ******************************************************
126 REM
127 CLS
128 PRINT "PROGRAMA DE CODIFICACION DE MENSAJES"
129 PRINT "===================================="
130 PRINT:PRINT
131 PRINT "INTRODUCE EL MENSAJE A CODIFICAR"
```

```
132 PRINT
133 LET N=VAL(MID$(TIME$,4,2))*ASC(MID$(TIME$,8,1))
134 RANDOMIZE N
135 INPUT A$
136 LET N=INT(RND*255)
137 LET A$=CHR$(N)+A$
138 LET B$=CHR$((N OR ASC(MID$(A$,2,1)))-(N AND ASC(MID$(A$,2,1))))
139 FOR I=2 TO LEN(A$)-1
140    LET N1=ASC(MID$(A$,I,1))
141    LET N2=ASC(MID$(A$,I+1,1))
142    LET B$=B$+CHR$((N1 OR N2)-(N1 AND N2))
143 NEXT I
144 LET B$=CHR$(N)+B$
145 CLS
146 PRINT
147 PRINT "CODIFICACION REALIZADA"
148 PRINT
149 PRINT "INTRODUCE ESTA LINEA EN EL PROGRAMA No. 3"
150 PRINT
151 PRINT "1000 DATA ";
152 PRINT LEN(B$);",";
153 FOR I=1 TO LEN(B$)
154    PRINT ASC(MID$(B$,I,1));
155    IF I<>LEN(B$) THEN PRINT ",";
156 NEXT I
157 PRINT
```

Este programa funciona perfectamente en el IBM y compatibles. Para todos los demás ordenadores las variaciones que hay que realizar son:

COMMODORE

127 PRINT "<SHIFT-HOME>"
134 LET N=RND(–TI)
136 LET N=INT(RND*255)
145 PRINT "<SHIFT-HOME>"

Hay que quitar la línea 133.

AMSTRAD

134 LET N=RND(–TIME)

Hay que quitar la línea 133.

MSX

134 LET N=RND(–TIME)
136 LET N=INT(RND(1)*255)

Hay que quitar la línea 133.

SPECTRUM

134 RANDOMIZE
138 LET B$=CHR$((N OR CODE(A$(2)))–(N AND CODE(A$(2))))
140 LET N1=CODE(A$(I))
141 LET N2=CODE(A$(I+1))
154 PRINT CODE(B$(I));

Hay que quitar la línea 133.

El funcionamiento de este programa línea a línea es el siguiente:

Línea 127. Borra la pantalla.
Línea 128. Imprime el nombre del programa.
Línea 129. Y lo subraya.
Línea 130. Deja una línea en blanco.
Línea 131. Imprime un mensaje que le dice al usuario que introduzca el mensaje a codificar.
Línea 132. Deja una línea en blanco.
Línea 133. Esta línea sólo es válida para los usuarios del IBM y compatibles. Calcula un número aleatorio basándose en el tiempo transcurrido desde la puesta en marcha del ordenador.
Línea 134. Inicializa el generador de números aleatorios. Esto se hace así para que el número clave para codificar no sea siempre el mismo.
Línea 135. Recoge del teclado el mensaje a codificar. Esto puede cambiarse por:

135 LET A$="SUPERCALIFRAGILISTICOES-PIALIDOSO

suponiendo que SUPERCALIFRAGILISTICOESPIALIDOSO fuese la clave de acceso al programa.

Línea 136. Se almacena en N un número aleatorio entre 0 y 254. Este es el número que se utilizará para la codificación.

Línea 137. Se une el carácter CHR$(N) con el mensaje a codificar.

Línea 138. Se codifica la primera letra del mensaje mediante la función XOR.

Línea 139. Empieza un bucle en el cual se codificarán todas las letras del mensaje empezando por la segunda.

Línea 140. En N1 se almacena el carácter al que apunta la variable I.

Línea 141. En N2 se almacena el carácter al que apunta I+1.

Línea 142. Se hace una operación XOR entre N1 y N2 y el resultado, después de convertirlo en un carácter, se almacena en B$.

Línea 143. En esta línea termina el bucle.

Línea 144. Se almacena al principio del mensaje codificado (B$) el valor de N.

Línea 145. Borramos la pantalla.

Líneas 146 a 150. Se imprime un mensaje.

Línea 151. Imprimimos el string 1000 DATA.

Línea 152. Se imprime la longitud de B$.

Línea 153. Aquí empieza un bucle dentro del cual se imprimirán, uno a uno, los códigos ASCII de todos los caracteres que se encuentran almacenados en la variable alfanumérica B$.

Línea 154. Se imprime el código ASCII de cada carácter de B$.

Línea 155. Si es el carácter impreso no es el último se imprime una coma.

Línea 156. Termina el bucle.

Línea 157. Se imprime una línea en blanco.

Una vez conocido el funcionamiento del programa vamos a explicar el algoritmo utilizado para codificar el mensaje.

Dicha codificación consiste en generar un carácter aleatorio y hacer una operación XOR con el primer carácter del mensaje. Una vez hecho esto, se coge el número que nos ha salido, lo almacenamos y se realiza la misma operación pero con el segundo carácter. El número resultante es almacenado y operado con el tercer carácter del mensaje. Esta operación se realiza hasta que hemos operado con el último de los caracteres.

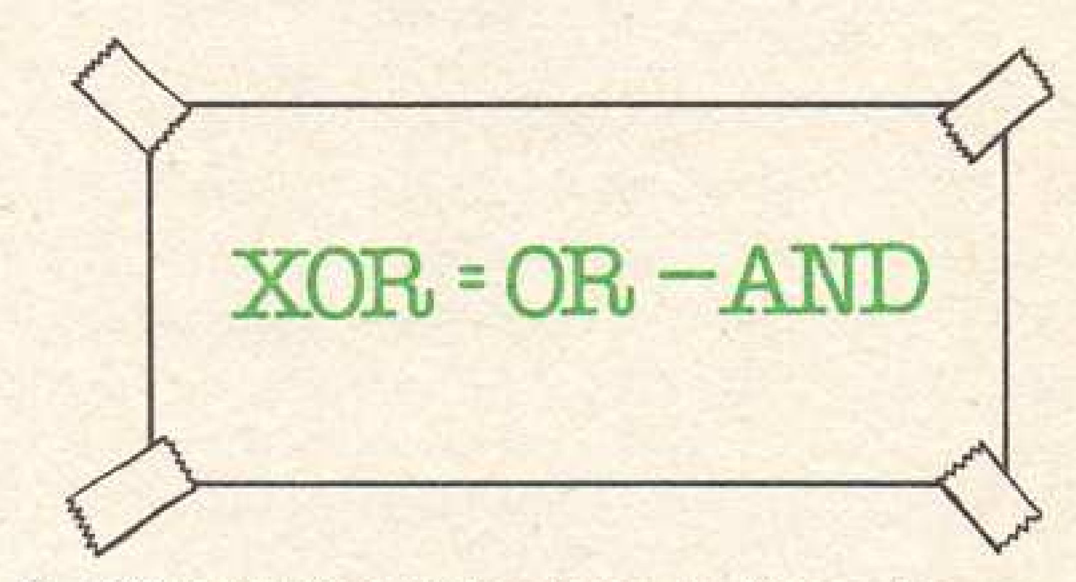


Fig. 5. Como muchos ordenadores no tienen incorporada la operación XOR nosotros la hemos simulado gracias a una operación OR seguida de AND.

El resultado de todo esto es un mensaje codificado que es imposible de entender si no se conoce el número generado aleatoriamente. Por ello, es necesario almacenarlo junto con el mensaje codificado.

El programa que se da a continuación es el inverso del que acabamos de ver. Este programa está preparado para unirlo con nuestros propios programas y así protegerlo más de las miradas indiscretas.

```
100 REM **********************************************************
101 REM * <<< DECODIFICADOR DF MENSAJES POR EL METODO XOR >>> *
102 REM *                                                    *
103 REM * VALIDO PARA AMSTRAD, IBM, MSX, COMMODORE Y SPECTRUM *
104 REM *                                                    *
105 REM **********************************************************
106 REM *                                                    *
107 REM * VARIABLES QUE HAY QUE PASARLE A LA RUTINA          *
108 REM * ---------------------------------------------      *
109 REM *                                                    *
110 REM *  NINGUNA. EL MENSAJE CODIFICADO SE ENCUENTRA EN LA  *
111 REM * SENTENCIA DATA DE LA LINEA 1000                    *
```

```
112 REM *                                                    *
113 REM * EL PROGRAMA NO DEVUELVE NINGUN VALOR               *
114 REM *                                                    *
115 REM * VARIABLES USADAS INTERNAMENTE                      *
116 REM * ----------------------------                       *
117 REM *                                                    *
118 REM * A$ = ALMACENA EL MENSAJE DECODIFICADO               *
119 REM * B$ = ALMACENA EL MENSAJE CODIFICADO                 *
120 REM * N  = NUMERO ALEATORIO                              *
121 REM * I  = CONTADOR DE BUCLE                             *
122 REM * N1 = VARIABLE AUXILIAR                             *
123 REM * N2 = VARIABLE AUXILIAR                             *
124 REM *                                                    *
125 REM ******************************************************
126 REM
127 RESTORE 1000
128 LET B$=""
129 READ N
130 FOR I=1 TO N
131     READ N1
132     LET B$=B$+CHR$(N1)
133 NEXT I
134 LET A$=MID$(B$,1,1)
135 FOR I=1 TO LEN(B$)-1
136     LET N1=ASC(MID$(A$,I,1))
137     LET N2=ASC(MID$(B$,I+1,1))
138     LET N=(N1 OR N2)-(N1 AND N2)
139     LET A$=A$+CHR$(N)
140 NEXT I
141 LET B$=MID$(A$,2)
142 LET A$=""
143 CLS
144 PRINT "POR FAVOR. INTRODUZCA EL CODIGO DE ACCESO"
145 PRINT:PRINT "---> _";CHR$(29);
146 FOR I=1 TO LEN(B$)
147     LET C$=INKEY$:IF C$="" THEN GOTO 147
148     PRINT CHR$(65+RND*25);"_";CHR$(29);
149     LET A$=A$+C$
150 NEXT I
151 CLS
152 PRINT "CODIGO DE ACCESO ERRONEO"
153 FOR I=1 TO 200:NEXT I
154 IF A$<>B$ THEN GOTO 151
155 CLS
156 PRINT "CODIGO DE ACCESO CORRECTO"
157 PRINT
158 PRINT "PULSE 'C' PARA EMPEZAR LA EJECUCION DEL PROGRAMA"
159 LET C$=INKEY$:IF C$<>"C" THEN GOTO 159
160 CLS
```

Las variaciones que hay que realizar para todos los ordenadores que no sean IBM o compatibles son los siguientes:

COMMODORE

```
143 PRINT "<SHIFT-HOME>"
154 PRINT "--->_";"<CURSOR IZQUIERDA>";
147 GET C$:IF C$="" THEN GOTO 147
148 PRINT CHR$(65+RND(1)*25);"_";"<CURSOR IZQUIERDA>";
151 PRINT "<SHIFT-HOME>"
155 PRINT "<SHIFT-HOME>"
159 GET C$:IF C$<>"C" THEN GOTO 159
160 PRINT "<SHIT-HOME>"
```

AMSTRAD

```
148 PRINT CHR$(65+RND(1)*25);"_";CHR$(29);
```

MSX

```
148 PRINT CHR$(65+RND(1)*25);"_";CHR$(29);
```

SPECTRUM

```
134 LET A$=B$(1)
136 LET N1=CODE(A$(I))
137 LET N2=CODE(B$(I+1))
141 LET B$=A$(2 TO)
145 PRINT "--->_";CHR$(8);
148 PRINT CHR$(65+RND*25);"_";CHR$(8);
```

Como puedes apreciar, las líneas comprendidas entre la 143 y la 160 son exactamente iguales que el programa n.° 1. Para que no lo tengas que copiar dos veces te recomiendo que hagas un MERGE y cambies todos los GOTOs.

El funcionamiento del programa hasta la línea 142 es el siguiente:

Línea 127. En esta línea le decimos al ordenador que los datos que vamos a leer a continuación se encuentran en el DATA de la línea 1000. Esta línea es la que nos imprimió en pantalla el programa n.° 2. En ella se encuentran los códigos necesarios para descifrar el mensaje. Antes de ejecutar este programa tenemos que introducir la línea 1000 tal y como nos la devolvió el programa n.° 2.

Línea 128. Inicializamos la variable B$.

Línea 129. Leemos el número de valores que tenemos que leer de la línea DATA.

Línea 130. Comenzamos un bucle desde 1 hasta N en el que iremos leyendo todos los números de la DATA.

Línea 131. Leemos un número (N1).

Línea 132. Almacenamos dicho número en B$.

Línea 133. Aquí termina el bucle.

Línea 134. Almacenamos en A$ el primer carácter de B$. El código ASCII de este carácter es el número aleatorio generado con el programa n.° 2 y que es la clave necesaria para descifrar el mensaje.

Línea 135. A partir de esta línea comienza un bucle desde 1 hasta la longitud de B$ menos uno (LEN(B$)-1) en el que se descifrará el mensaje.

Línea 136. Se almacena en la variable N1 el código ASCII de la letra que apunta la variable I. En la primera vuelta del bucle este valor es el número necesario para descifrar la clave. A partir de la segunda vuelta, dicho número es el código ASCII de una de las letras del mensaje.

Línea 137. Se almacena en la variable N2 el valor del siguiente carácter a decodificar.

Línea 138. Se hace la operación XOR entre N1 y N2. El resultado se almacena en N.

Línea 139. Se concatena A$ con el último carácter decodificado.

Línea 140. Aquí termina el bucle.

Línea 141. Se asigna a B$ todos los caracteres almacenados en A$ menos el primero. El primero es el carácter cuyo código ASCII nos ha servido para decodificar el mensaje, pero que no forma parte de él.

Línea 142. Se reinicializa la variable numérica A$.

A partir de esta línea el programa es igual que el n.° 1.

Puede que te estés preguntando para qué sirve todo esto si no conocemos la forma de hacer que nuestros programas no puedan ser protegidos. Todo esto sirve para que tú vayas tomando práctica en la codificación de mensajes. Más adelante veremos cómo proteger nuestros programas de las miradas indiscretas. También veremos cómo protegerlos contra las copias piratas.

En el siguiente apartado te propongo algunos ejercicios para que tú mismo encuentres distintas formas de codificación. También te recomiendo el libro:

CRIPTOGRAFIA. LA OCULTACIÓN DE MENSAJES Y EL COMPUTADOR

de Vicente Martínez Orga y que está editado por Ediciones Siglo Cultural, dentro de su colección: INFORMATICA APLICADA.

Como el tema es muy amplio, y muy interesante, en sucesivos tomos volveremos a tratar sobre el tema con nuevas rutinas y ejercicios de codificación de mensajes.

Ejercicios resueltos

EJERCICIO N.° 1

El método de codificación que hemos visto es uno de los más sencillos. ¿Serías capaz de realizar un programa que codificase y decodificase por el método de TRANSPOSICION?

Este método consiste en asignar a cada letra del abecedario otra letra del mismo, pero con una traslación de N letras. Por ejemplo, si tenemos que el alfabeto es:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

y lo escribimos de nuevo con una traslación de 10 nos daría:

J K L M N O P Q R S T U V W X Y Z A B C D E F G H I

si quisiéramos codificar el mensaje miraríamos letra a letra su posición en la primera línea, nos bajaríamos a la segunda y escribiríamos la letra que aparece en esta última. Según esto, si tenemos la palabra clave:

CHURRIGUERESCO

ésta se convertiría en:

LQDAARGDNANBLX

¿Serías capaz de hacer un programa que cifrase por el método de transposición con sólo darle el mensaje y el número de traslación? ¿Y que descifrase dándole el mensaje cifrado y el número de traslación?

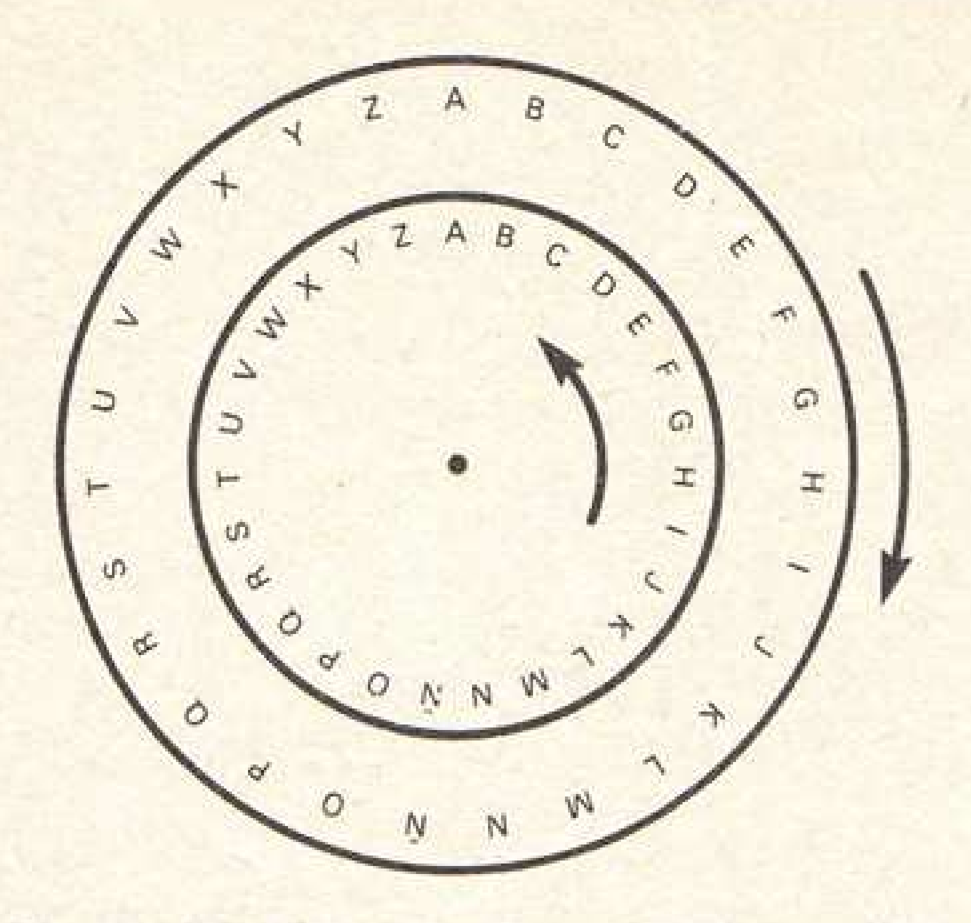


Fig. 6. Para realizar el método de codificación por transposición se pueden utilizar dos discos de cartón unidos por el centro y capaces de girar uno sobre otro.

SOLUCION

```
1000 REM **********************************************************
1001 REM *  <<< CODIFICACION POR EL METODO DE TRANSPOSICION >>>  *
1002 REM *                                                        *
1003 REM *  VALIDO PARA MSX, IBM, AMSTRAD, COMMODORE Y SPECTRUM  *
1004 REM *                                                        *
1005 REM **********************************************************
1006 REM
1007 CLS
1008 PRINT
1009 PRINT "CIFRADO POR TRANSPOSICION"
1010 PRINT "------------------------"
1011 PRINT
1012 PRINT "INTRODUCE EL MENSAJE A CODIFICAR"
1013 PRINT
1014 INPUT M$
1015 LET Z$=" 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz?',.
;:-+=><"
1016 PRINT
1017 PRINT "INTRODUCE TRASLACION (ENTRE 1 Y 74)"
1018 PRINT
1019 INPUT T
1020 IF T<1 OR T>74 THEN PRINT:PRINT "TRASLACION ERRONEA":GOTO 1016
1021 LET A$=""
1022 FOR I=1 TO LEN(M$)
1023    LET N=INSTR(Z$,MID$(M$,I,1))+T
1024    IF N>74 THEN LET N=N-74
1025    LET A$=A$+MID$(Z$,N,1)
1026 NEXT I
1027 CLS
1028 PRINT "EL MENSAJE:":PRINT
1029 PRINT M$
1030 PRINT
1031 PRINT "CON UNA TRASLACION DE ";T
1032 PRINT
1033 PRINT "SE CONVIERTE EN:":PRINT
1034 PRINT A$
1035 PRINT
1036 END
```

El programa n.º 4 es una de las posibles soluciones que se nos pueden ocurrir para cifrar un mensaje.

Este programa funciona perfectamente en el IBM, AMSTRAD y MSX. Para el COMMODORE y SPECTRUM hay que realizar las siguientes modificaciones:

COMMODORE

1007 PRINT "<SHIFT-HOME>"
1023 GOSUB 2000
1027 PRINT "<SHIFT-HOME>"

También hay que añadir este grupo de líneas:

2000 FOR J=1 TO 74
2001 IF MID$(M$,I,1)=MID$(Z$,J,1) THEN N=J+T:J=74
2002 NEXT J
2003 RETURN

SPECTRUM

1023 GOSUB 2000
1025 LET A$=A$+Z$(N)

También hay que añadir las siguientes líneas:

2000 FOR J=1 TO 74
2001 IF M$(I)=Z$(J) THEN LET N=J+T:LET J=74
2002 NEXT J
2003 RETURN

A continuación aparece el programa que decodifica el mensaje.

```
1000 REM *********************************************************
1001 REM * <<< DECODIFICACION POR EL METODO DE TRANSPOSICION >>> *
1002 REM *                                                       *
1003 REM *  VALIDO PARA MSX, IBM, AMSTRAD, COMMODORE Y SPECTRUM   *
1004 REM *                                                       *
1005 REM *********************************************************
1006 REM
1007 CLS
1008 PRINT
1009 PRINT "DESCIFRADO POR TRANSPOSICION"
1010 PRINT "----------------------------"
1011 PRINT
1012 PRINT "INTRODUCE EL MENSAJE A DECODIFICAR"
1013 PRINT
1014 INPUT M$
1015 LET Z$=" 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz?!,.
;:-+=><"
1016 PRINT
1017 PRINT "INTRODUCE TRASLACION (ENTRE 1 Y 74)"
1018 PRINT
1019 INPUT T
1020 IF T<1 OR T>74 THEN PRINT "TRASLACION ERRONEA":GOTO 1016
1021 LET A$=""
1022 FOR I=1 TO LEN(M$)
1023     LET N=INSTR(Z$,MID$(M$,I,1))-T
1024     IF N<1 THEN LET N=74
1025     LET A$=A$+MID$(Z$,N,1)
1026 NEXT I
1027 CLS
1028 PRINT "EL MENSAJE CODIFICADO:":PRINT
1029 PRINT M$
1030 PRINT
1031 PRINT "CON UNA TRASLACION DE ";T
1032 PRINT
1033 PRINT "UNA VEZ DECODIFICADO SE CONVIERTE EN:":PRINT
1034 PRINT A$
1035 PRINT
1036 END
```

Las modificaciones que hay que realizar para el COMMODORE Y SPECTRUM son las que se dan a continuación.

COMMODORE

```
1007 PRINT "<SHIFT-HOME>"
1023 GOSUB 2000
1027 PRINT "<SHIFT-HOME>"
```

Como en el programa anterior, hay que añadir las siguientes líneas:

```
2000 FOR J=1 TO 74
2001 IF MID$(M$,I,1)=MID$(Z$,J,1) THEN
N=J-T:J=74
2002 NEXT J
2003 RETURN
```

SPECTRUM

```
1023 GOSUB 2000
1025 LET A$=A$+Z$(N)
```

Añadiendo las siguientes líneas:

```
2000 FOR J=1 TO 74
2001 IF M$(I)=Z$(J) THEN LET N=J-T:LET J=74
2002 NEXT J
2003 RETURN
```

EJERCICIO N.° 2

¿Serías capaz de modificar los dos programas anteriores para que, aparte de codificar, almacenasen el mensaje al contrario? Por ejemplo, el mensaje ADIOS, codificado con el programa n.° 4, se nos convierte en BEJPT. Al darle la vuelta nos aparecería como:

TPJEB

SOLUCION

En el programa n.° 4 lo único que hay que hacer es cambiar la línea 1025 por:

```
1025 LET A$=MID$(Z$,N,1)+A$
```

Los usuarios del SPECTRUM tendrían que poner:

```
1025 LET A$=Z$(N)+A$
```

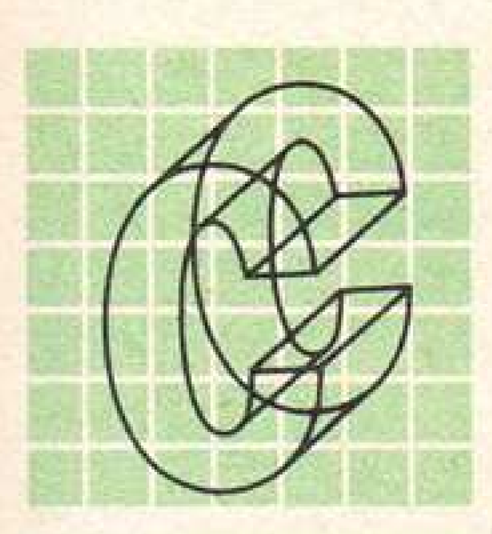
En el programa de decodificación habría que cambiar la línea 1025 por:

```
1025 LET A$=MID$(Z$,N,1)+A$
```

y en el SPECTRUM por:

```
1025 LET A$=Z$(N)+A$
```

HERRAMIENTAS DEL TALLER DEL HARDWARE



ON los problemas enunciados al proponer la tarjeta de ampliación de puertos para nuestro ordenador personal se ve la conveniencia de detallar las herramientas más idóneas de las que debería constar nuestro taller. Muchas de ellas, con seguridad, están ya entre las de uso común para cualquier actividad casera, pero conviene mencionarlas todas por si se nos han pasado o no hemos encontrado la necesidad de disponer de ellas.

Empezaremos por dividirlas en grupos: mecánicas, eléctricas-electrónicas y de organización.

Herramientas mecánicas

El trabajo de electrónica que acompaña a la informática aplicada tiene una componente que podríamos decir menos noble y que por ello a veces se minusvalora en cuanto a su utilidad. El desarrollo de cualquier proyecto relacionado con periféricos requerirá, sin duda alguna, el empleo de las herramientas típicas de taller, a menudo denominadas "quincalla", para desmontar, ajustar o poner a punto los equipos conectados. Describimos las más usuales y daremos indicaciones de cómo seleccionarlas de las que con abundancia nos ofrecen los suministradores del ramo.

Alicates: Es conveniente disponer de varios tipos:

• De puntas, para conformado de cables y chapas de conexión.

• De corte, para cortar los cables y los terminales soldados. Conviene que sean de puntas pequeñas y afilables, para poder dar cortes nítidos sin forzar el terminal.

• Normal, para el trabajo típico de sujeción de elementos.

Pinzas

Para poder acceder a componentes que se caen dentro del equipo, sin necesidad de desmontar todo él. También se emplean para sujetar la patilla de un transistor, al soldarle, para que haga de barrera térmica.

Otras pinzas de interés son las de toma de señal. Hay una gran variadad. Son recomendables las que permiten el contacto con una patilla sin tocar la próxima.

Destornilladores

Es conveniente disponer de un juego variado, para utilizar el más apropiado para cada tornillo, de manera que ninguno de ellos sufra en la operación. Existen juegos de destornilladores para pequeños tamaños con punta en acero tratado y que suelen emplearse en trabajos de relojería. Cada vez es más corriente encontrarse con equipos que poseen tornillos con cabeza en cruz, los cuales deben ser atornillados con un destornillador de estrella. Si el par de apriete no es muy fuerte, pueden desatornillarse con un destornillador plano, pero lo normal es que las puntas del destornillador queden "resentidas".

Llaves

Los tornillos de ajuste de los equipos mecánicos suelen ser de cabeza hexagonal interior, siendo necesario el empleo de llaves Allen o de un tipo parecido con las puntas más agudas.

No debería faltar una pequeña llave inglesa, para poder apretar y aflojar cualquier

tornillo de cabeza hexagonal, sin deterioro, cosa que seguramente ocurriría si utilizáramos alicates.

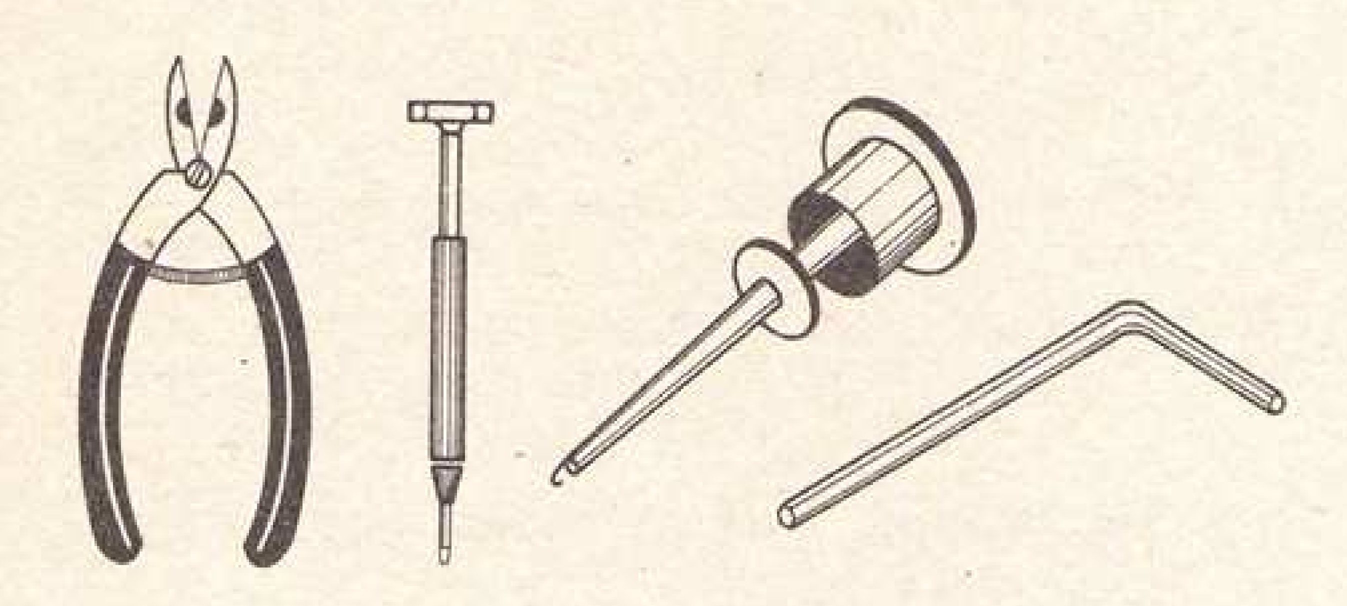


Fig. 1. Alicates de corte, destornillador, pinzas y llave Allen.

Sierras

Para el trabajo de chapa o preparación de cajas para montar los equipos será necesario utilizar primero la taladradora, manual o mecánica, y luego sierra o lima para ajustar el tamaño a los conectores y demás piezas auxiliares. Las sierras necesarias son:

- De marquetería, para corte de chapa de aluminio, plástico, circuitos impresos y chapas finas en general. Permite un acabado fino si se sabe emplear sin romper la segueta. Hay diferentes tipos de segueta según el material. Es fundamental que la pieza a cortar esté bien sujeta al tornillo del banco o a la mesa mediante una mordaza.
- De arco, para las piezas de acero, ejes de potenciómetros, etc.

Limas

Para el acabado y ajuste de las piezas mecánicas será necesario disponer de algunas limas pequeñas, de las denominadas de relojero. Conviene tener de tipo plano, media caña y triangular, por lo menos.

Tornillo de banco

Es una pieza fundamental para cualquier trabajo mecánico. Debe permitir sujetar la pieza en la posición apropiada para el trabajo que se necesite. Hay muchas variedades, entre las que resulta conveniente el tipo de mordaza orientable, mediante una rótula que la une a la mesa.

Brocas

Para cualquier tipo de perforación es necesario disponer de una taladradora, a ser posible eléctrica. Para los trabajos típicos de electrónica existen unas pequeñas, alimentadas en continua, con lo que puede ajustarse la velocidad y que permiten el taladrado de los circuitos impresos de forma óptima. Por supuesto, las taladradoras manuales permiten a la vez hacer gimnasia de brazos.

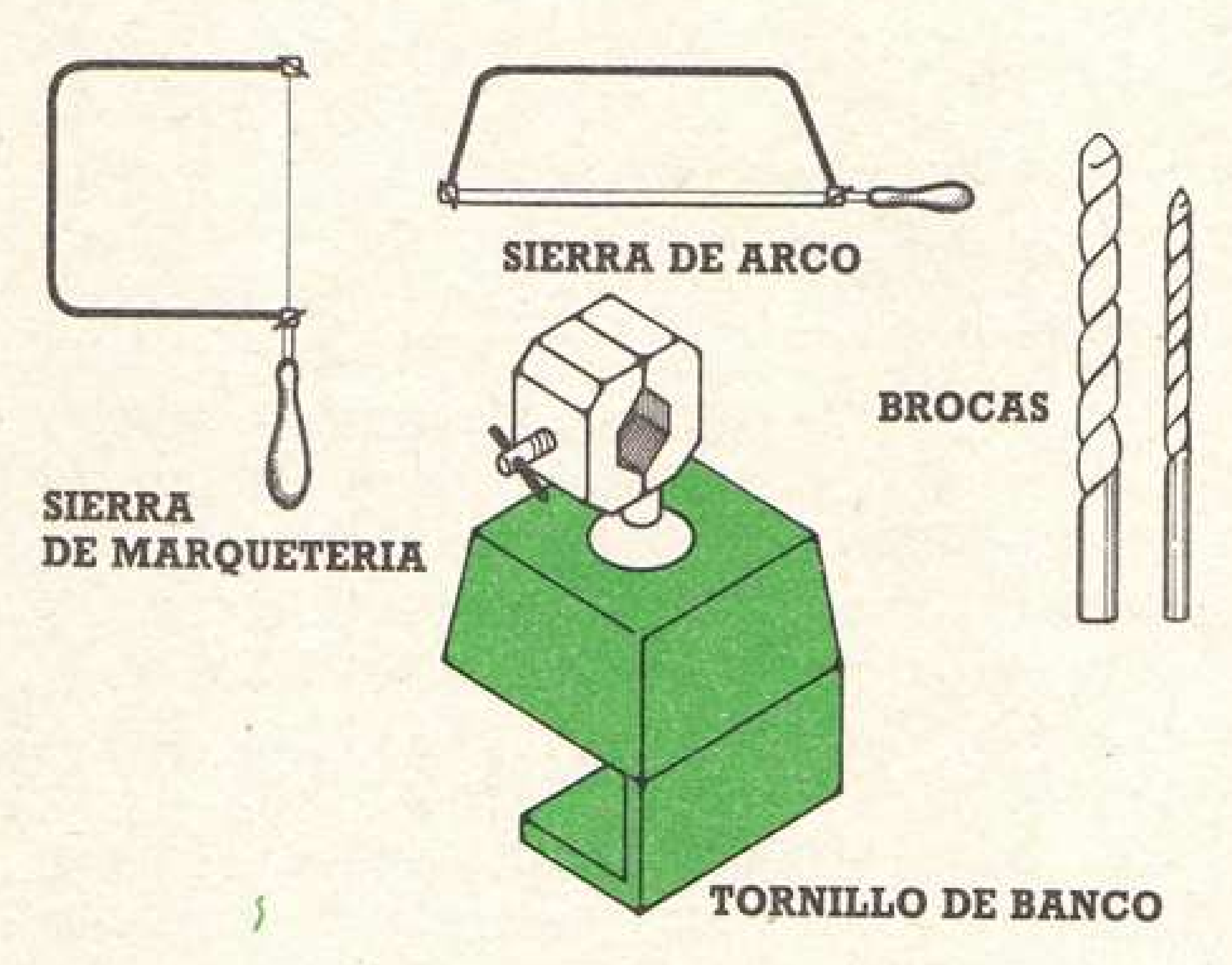


Fig. 2. Sierras, limas, tornillo de banco y brocas.

Cortadores

Bajo este nombre agrupamos a todas las herramientas que sirven para cortar, como su nombre indica: cuchillas de diferentes formas, pelacables, tijeras.

Las cuchillas tanto de afeitar como las de mango en forma de lápiz, sirven para reparar los circuitos impresos y realizar cortes precisos en plástico. Las hay de formas muy variadas, permitiendo cortes como de bisturí.

Para pelar los cables hay muchas variantes, pareciendo todas ellas como tijeras modificadas. El más sencillo y efectivo realiza el corte mediante dos hendiduras afiladas en dos piezas en ángulo, que puede apretarse hasta un límite fijado por un tornillo. Permite cortar el aislamiento del cable sin dañar el cobre.

Manos de ayuda

Para el montaje de conectores y otros tipos de soldaduras delicadas existen unos dispositivos muy sencillos e ingeniosos que per-

miten sujetar varios componentes a la vez orientándolos adecuadamente para trabajar con ellos. Están compuestos de varias pinzas de cocodrilo sujetas a unas rótulas que pueden fijarse en cualquier posición. Es un "amigo" insustituible para cualquier trabajo que requiera mantener en posición más de dos piezas, mientras se utiliza simultáneamente el soldador y se aporta estaño. Solamente si se poseen los brazos de un pulpo se puede hacer algo similar.

Lupa

Para algunas soldaduras delicadas y en general para determinar posibles puntos de ruptura en circuitos impresos es conveniente una lupa de 6 a 8 aumentos. En algunas "manos de ayuda" se puede montar, también mediante rótulas una lupa, con lo que se puede trabajar con mayor precisión. A veces conviene observar la parte de atrás de componentes sin tener que desmontarlos. Para ello se emplean unos espejitos en el extremo de una varilla, que pueden orientarse en cualquier dirección. Son como los espejos de dentista.

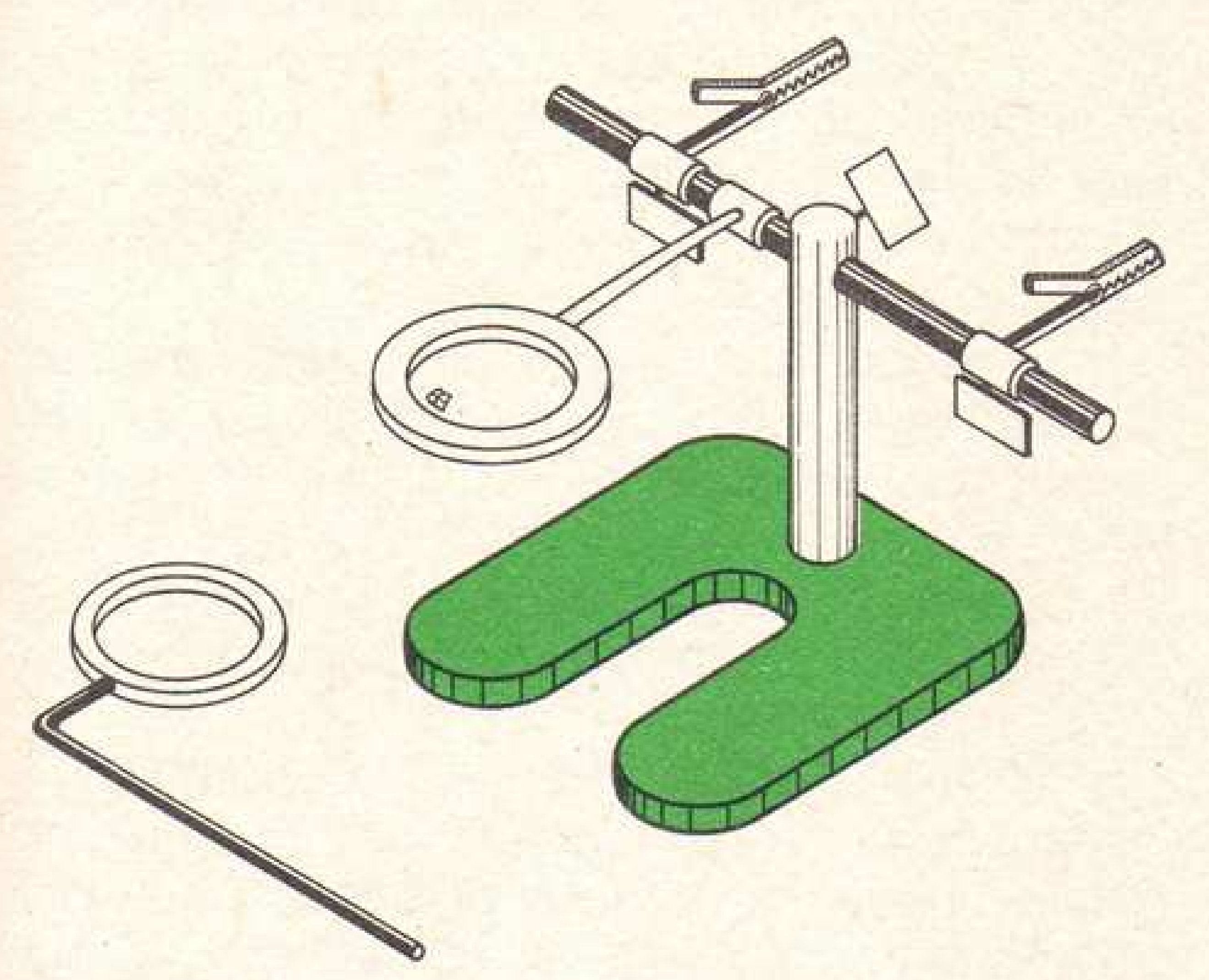


Fig. 3. Manos de ayuda con lupa y espejo.

Adhesivos

Es conveniente disponer siempre de varios tipos de adhesivos.

• Tipo ultrarrápido con Cyanolit, para sujeción de piezas de poca solicitud de esfuerzo.

• Tipo epoxy, para sujeción de piezas de bastante solicitud de esfuerzo. Requieren normalmente varias horas hasta poder utilizar la unión pegada. El más popular es el Araldit, pero existen muchos otros de características similares.

Cinta aislante

Para cualquier empalme es necesario recubrir con cinta aislante de PVC o Teflón. Existen actualmente unos tubos de plástico termorretráctil que permiten un acabado óptimo de cualquier empalme. Hay de diferentes diámetros. Se corta una longitud algo superior al empalme a recubrir, se introduce por uno de los extremos y se coloca sobre la zona. Al aplicar muy cerca el soldador, sin llegar a tocar, veremos que se contrae hasta sujetar y cubrir totalmente la unión. Industrialmente se aplica aire caliente, con lo que la terminación es más uniforme.

Herramientas eléctricas-electrónicas

Mencionamos todas las que se utilizan en cualquier taller de aficionado, y que tienen aplicación también en otros campos relacionados con la electrónica.

Polímetro

También se denominan a veces multímetros o Volt-Ohm-In. Es la pieza fundamental para cualquier tipo de ensayo o diseño. Además, los precios son asequibles a cualquier bolsillo, si nos conformamos con prestaciones limitadas. Siempre el último grito hay que pagarlo, como cualquier cosa de moda.

Hay que distinguir los equipos analógicos o de aguja de los digitales.

Los analógicos se basan en el movimiento de una bobina dentro de un campo magnético uniforme. Por tanto, su sensibilidad depende de la calidad del equipo de la bobina móvil. Conviene utilizar uno de por lo menos 20.000 ohmios por voltio. Esta forma de definir la sensibilidad quiere decir que hay que poner en serie con el instrumento una resistencia de 20.000 ohmios por cada voltio para conseguir que alcance el fondo de la escala. Otra forma de decirlo sería la intensidad necesaria para alcanzar el fondo de la escala. Para uno de 20.000 ohmios por voltio la intensidad es de 50 microamperios.

Los digitales, cada vez más económicos y robustos, presentan el resultado de las medidas en forma numérica, con indicación de las magnitudes, rango y otras variables asocia-

das a la medida. La sofisticación y economía de escala ha permitido que por el precio de un polímetro se pueda conseguir una verdadera unidad de medidas, que realiza de forma programable medidas en secuencia, selección automática de rango, promedios, gráficas, detección de alarmas, almacenamiento, comunicación con un ordenador y un sinfín de funciones.

Los tipos de medida de que pueden disponer son:

- Voltios continua. Márgenes con fin de escala en 1-10-100-1.000 voltios
- Voltios alterna. Márgenes iguales que en continua. Normalmente la sensibilidad es menor en alterna en los analógicos.
- Amperios continua. Márgenes: 1-10-100-1.000 miliamperios
- Amperios alterna. Igual que para continua.
- Resistencia. Márgenes × 1- × 10- × 100- × 1.000 ohms.
- Capacidades. Los condensadores grandes pueden medirse por el tiempo de carga, a partir de una tensión fija. Se suele utilizar la conexión de medida de resistencias. Los condensadores bajos pueden medirse como resistencias con alimentación en alterna.

De cara a una elección para nuestro taller, conviene tener en cuenta nuestro presupuesto, pero es recomendable un equipo digital de por lo menos tres dígitos y medio, que es como querer decir que pueda dar las medidas con tres cifras significativas o 4 si la primera es el 1.

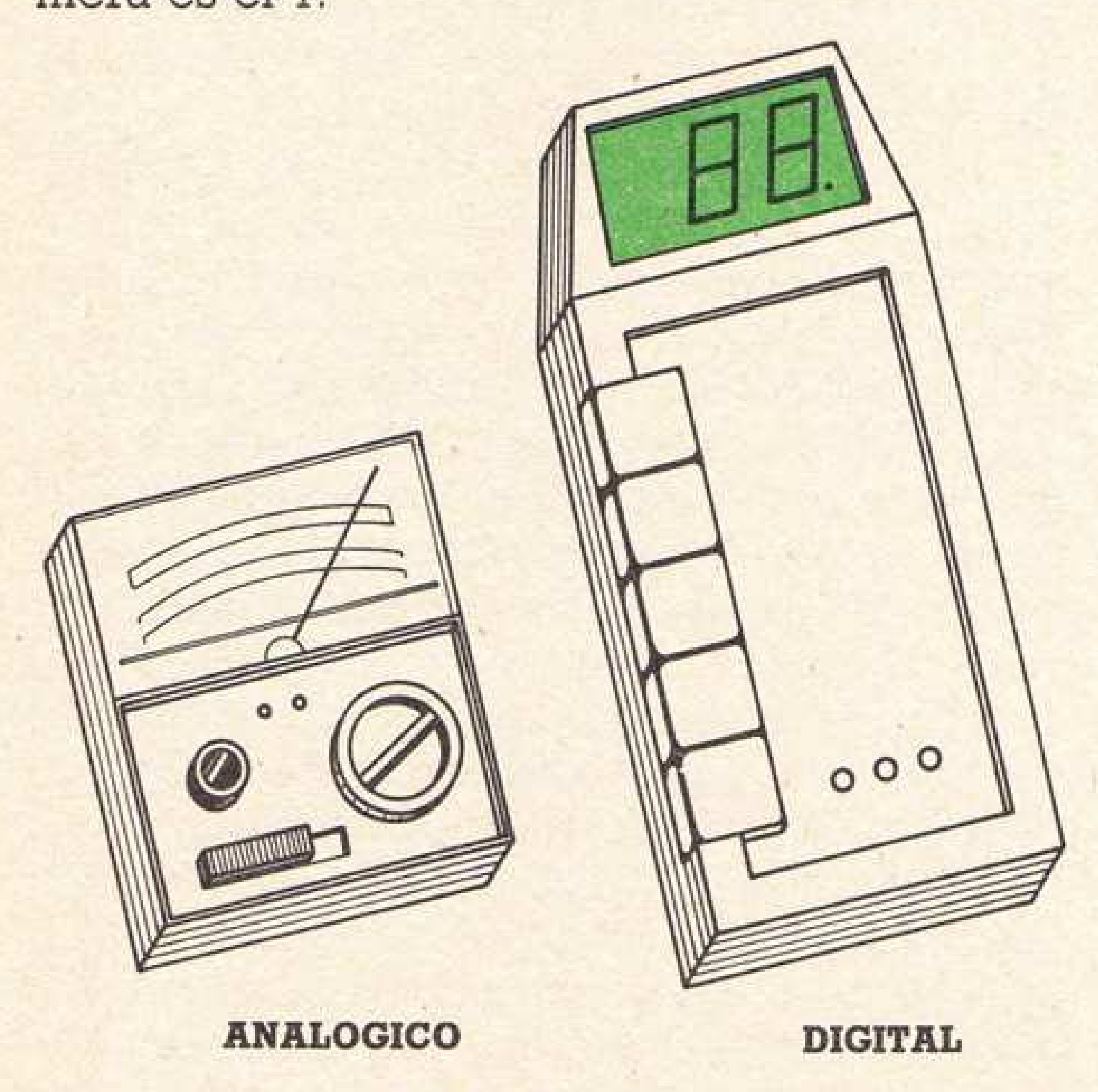


Fig. 4. Polímetros.

Enchufe múltiple

Es conveniente disponer de unos cuantos enchufes de corriente alterna con protección y filtro de ruidos. Nos dará seguridad en las medidas y reducirá las posibilidades de fallo. Podemos construirlo nosotros mismos, sobre una caja rectangular, con enchufes normales con toma de tierra, un interruptor diferencial y un filtro de red.

Soldador

Es un instrumento básico en cualquier taller de aficionado. El necesario para nuestras aplicaciones es cualquiera de poca potencia con la punta fina. De 15 vatios es lo ideal para circuitos integrados. Es todo un arte el soldar perfectamente en cualquier circunstancia, pero si seguimos los pasos recomendados, nos acercaremos a la perfección poco a poco. Como complemento hay que disponer de un soporte para el soldador que nos permita trabajar sin mover mucho el brazo y que lo mantenga en lugar seguro. En la misma base conviene disponer de una esponja mojada, sobre la que limpiaremos la punta. Es fundamental que la punta esté libre de residuos de resina y brillante en todo momento. Si después de mucho tiempo sin usar no conseguimos que brille, deberemos lijar suavemente, estando el soldador frío.

La casa BJC ofrece una gran variedad de equipos de soldadura, siendo muy interesantes las estaciones de soldadura, que incluyen: soporte, esponja y control de temperatura. Con el control podemos ajustar la potencia disipada según el tipo de componente que necesitemos soldar.

La forma de soldar es importante para un correcto funcionamiento de los equipos. Una excesiva temperatura puede deteriorar el componente, cambiando sus características o destruirlo definitivamente. Por el contrario, si la temperatura resulta insuficiente la soldadura puede quedar "fría", ocasionando fallos de forma intermitente. Podemos estimar la calidad de la soldadura por la apariencia externa: ha de ser brillante y con la forma de la figura. Si la gota de soldadura sobrepasa el contacto, es muy posible que la unión haya quedado fría. Es necesario que el punto a soldar se caliente por el soldador antes de aplicar el estaño. Es suficiente 2-3 segundos. Después el estaño fluirá por la unión, haciendo un contacto perfecto. Para la soldadura en componentes

grandes, será necesario calentar por más tiempo los contactos, antes de aplicar el estaño. Si lo que se suelda es un hilo a un terminal, es conveniente que el hilo haga forma de gancho. No es necesario que se retuerza sobre el terminal, pero sí que lo agarre en forma de gancho.

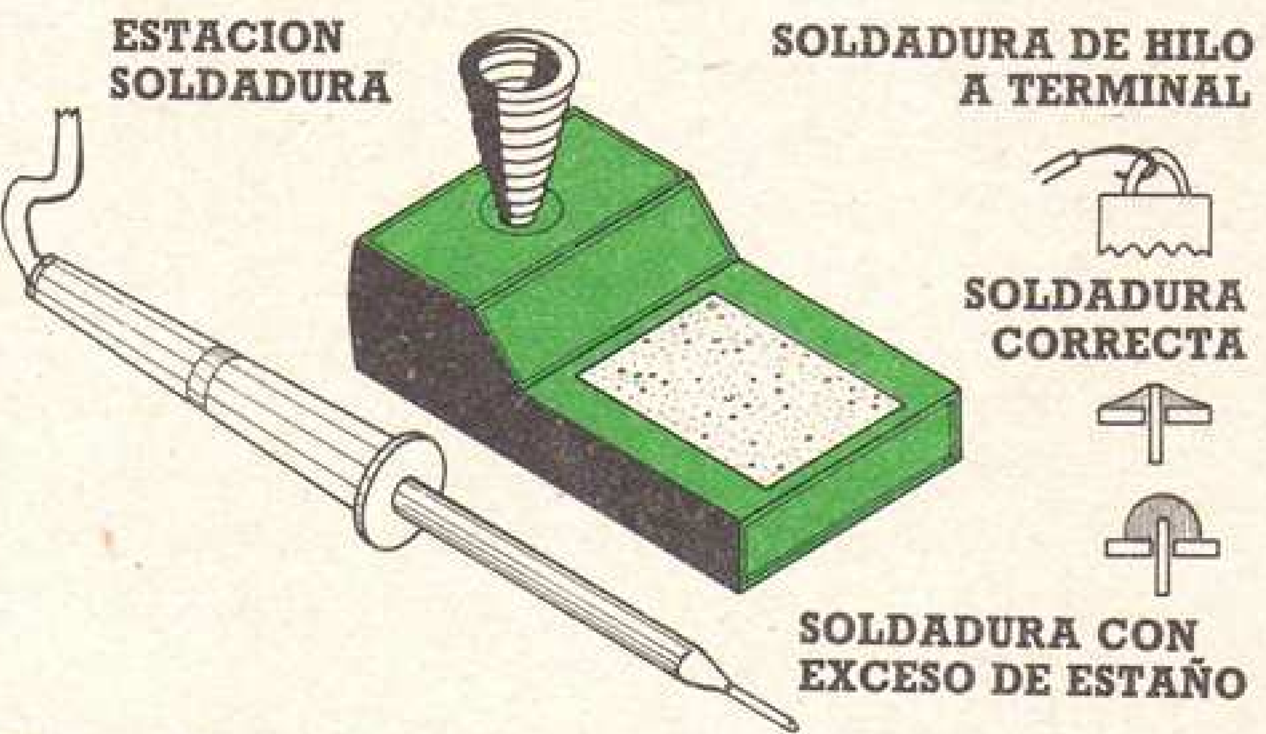


Fig. 5. Estación de soldadura, soldaduras de terminal y de hilo.

Estaño

Es necesario usar del tipo empleado en electrónica, con resina integrada internamente. Hay varios tipos, según la proporción de estaño/plomo que posea. El recomendado es de 60/40.

Desoldador

Es un instrumento asociado al soldador, para poder reparar un circuito o corregir una equivocación. Hay de varios tipos:

• De succión por émbolo. Se comprime un muelle y se expulsa el aire de un pistón terminado en punta. Al soltarlo, mientras se calienta el punto a desoldar se hace el vacío, extrayendo el estaño fundido. La punta es de Teflón para poder resistir la temperatura.

• De succión por perilla de goma. Se realiza un proceso similar, pero en este caso el aporte de calor a la unión lo hace el propio desoldador, cuya punta se mantiene a temperatura de fusión. Periódicamente es necesario limpiar la boquilla, pues con el estaño oxidado obstruye el paso del aire y con ello la capacidad de absorción.

• De mecha de cobre. Mediante una mecha realizada con hilos de cobre muy finos, se extrae por capilaridad el estaño, fundido en el punto de soldadura con un soldador.

Hay algunos instrumentos que se emplean para el proceso de desoldadura, como son:

• El soldador de patas múltiples, que permite calentar todas las patas de un circuito integrado.

• Las pinzas de extracción de circuitos. Permiten agarrar el circuito integrado por los extremos, mientras por el lado contrario se aplica el soldador de patas múltiples.

Sondas de prueba

Para el trabajo de diseño y verificación de lo diseñado es conveniente disponer de varias sondas que permitan ver el estado lógico de un punto de un circuito, mientras ejecutamos un programa. El ideal para este propósito es desde luego el osciloscopio, pero por su precio es conveniente "pasarse" sin él si es que no se puede adquirir. Con un circuito integrado y unos pocos LEDS podemos realizar un sustituto que en muchos caso puede darnos incluso más información que un osciloscopio.

Veamos diferentes tipos de sonda que se pueden montar:

• Sonda de estados lógicos. Permite ver el estado de un circuito de forma estática. Es decir, se encenderá si el estado es 1 y se apagará si es cero. Claro que si somos amigos de los colores podemos asignar un color al estado 1 y otro color al estado 0. Hay LEDS que pueden activarse en un color u otro. Para una primera versión es suficiente con asignar luz/no luz, según el estado. El circuito primero consta simplemente de unas puertas inversoras con histéresis. Al estado 1 de entrada dan estado 0, con lo que un LED conectado como se indica en la figura lucirá.

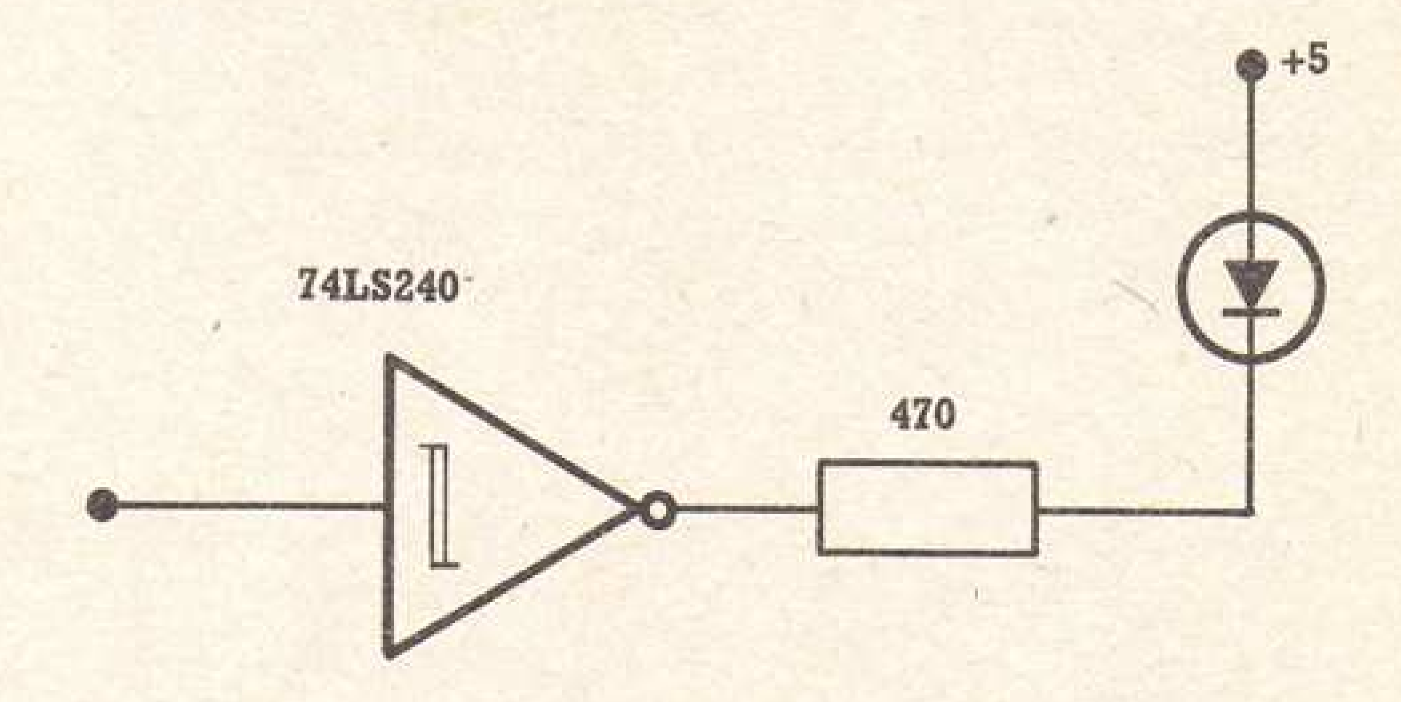


Fig. 6. Sonda de nivel.

La sonda no es todo lo ideal que desearíamos, pues carga al circuito que vamos a medir como una carga TTL-LS. Normalmente no será problema, pero es necesario tenerlo en cuenta. Podemos montar varias sondas de es-

tas, utilizando un solo circuito integrado, para poder observar varias señales de salida simultáneamente. Si distribuimos los LED linealmente sobre una carátula y sacamos hilo de entrada mediante una pinza, podemos asignar colores a cada una de las sondas para saber a qué señal corresponden.

Podemos hacer una sonda un poco más complicada pero que cargue mucho menos si colocamos amplificadores a la entrada, con transistores.

• Sonda de transiciones

Si además deseamos que nos indique la aparición de pulsos, aunque éstos sean de muy corta duración, deberemos añadir un circuito de temporización o monoestable, para

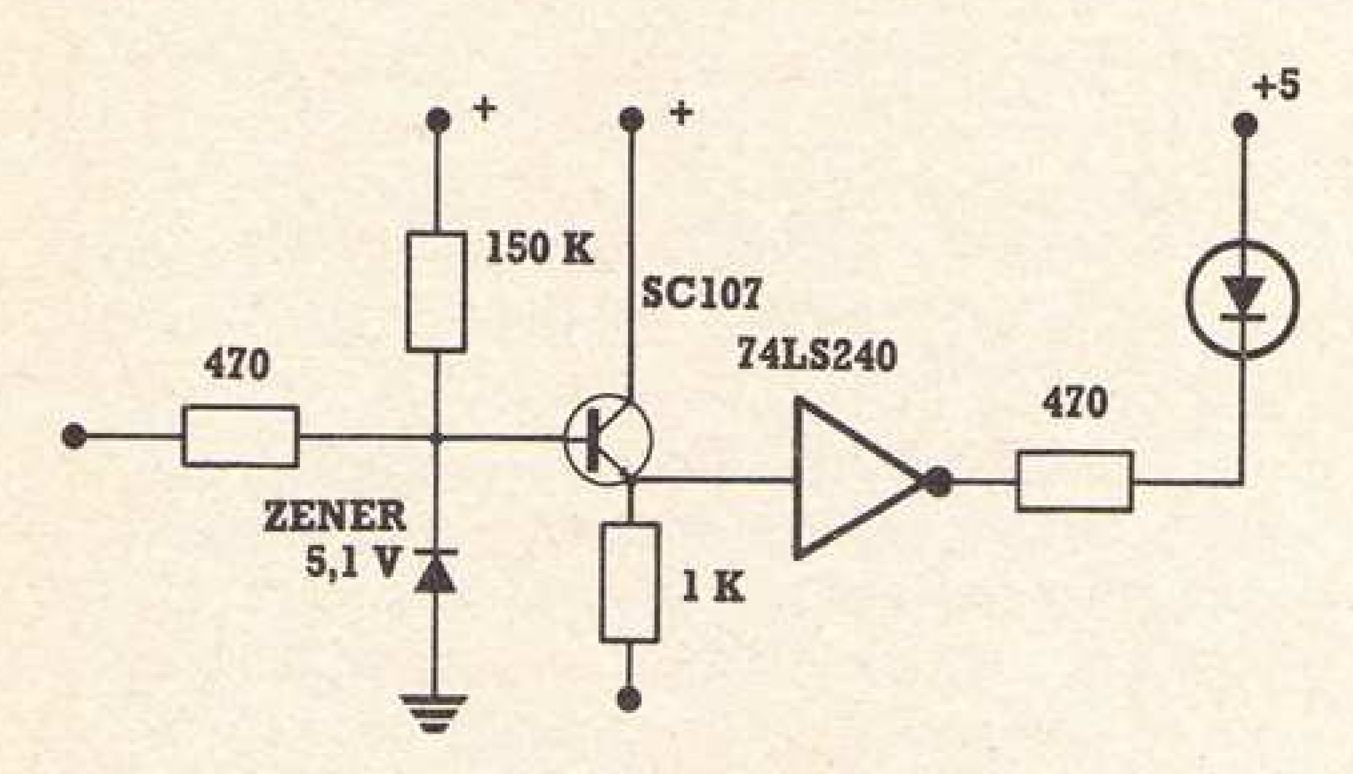


Fig. 7. Sonda con amplificación y protección.

que se active por el pulso y se mantenga unas décimas de segundo, para permitir su observación.

• Sonda de nivel variable.

Generalmente los equipos que vamos a observar o diseñar pertenecen a una familia de componentes TTL o CMOS. En los ordenadores personales lo más normal es que trabajen entre 0 y 5 voltios. Los valores a los cuales se produce la transición de estado 0 a estado 1 es diferente en cada tecnología.

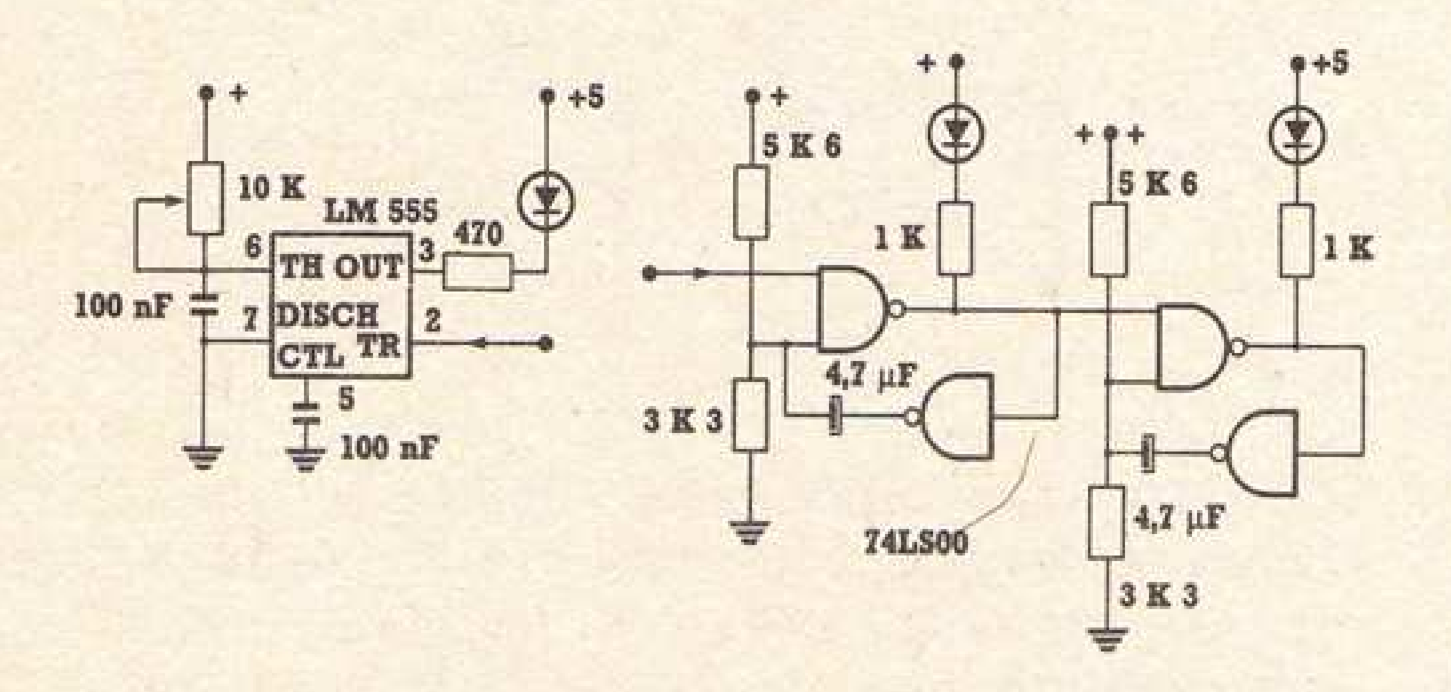


Fig. 8. Sonda de transiciones.

Para TTL el estado 0 corresponde a niveles de 0 a 0,8 voltios. El estado 1 corresponde a niveles de 2 a 5 voltios. Los niveles entre 0,8 y 2 voltios son niveles no válidos. Para un nivel de transición teórico de 1,4 voltios existe un margen de ruido admisible de +/- 0,6 voltios.

Para CMOS trabajando a 5 voltios, el estado 0 corresponde a niveles de 0 a 1,66 voltios, o más exactamente 1/3 de la tensión de alimentación. El estado 1 corresponde a 3,33, o más exactamente 2/3 de la tensión de alimentación. El punto de transición teórico es 1/2 de la tensión de alimentación.

Esta diferencia de niveles indica que la sonda debe conocer el punto de transición teórico, pues si no en algunos casos pudiera dar indicaciones erróneas.

Con el circuito de la figura podemos establecer el nivel de transición, con lo cual podemos adaptar al tipo de tecnología usada. El circuito es mucho más complicado que los anteriores y necesita un circuito integrado para cada señal binaria a observar.

• Sonda con retención

En determinados casos puede ser necesario saber si en un circuito se produjo un pulso, aunque fuera una sola vez. Para ello montamos un circuito como el de la figura, que al

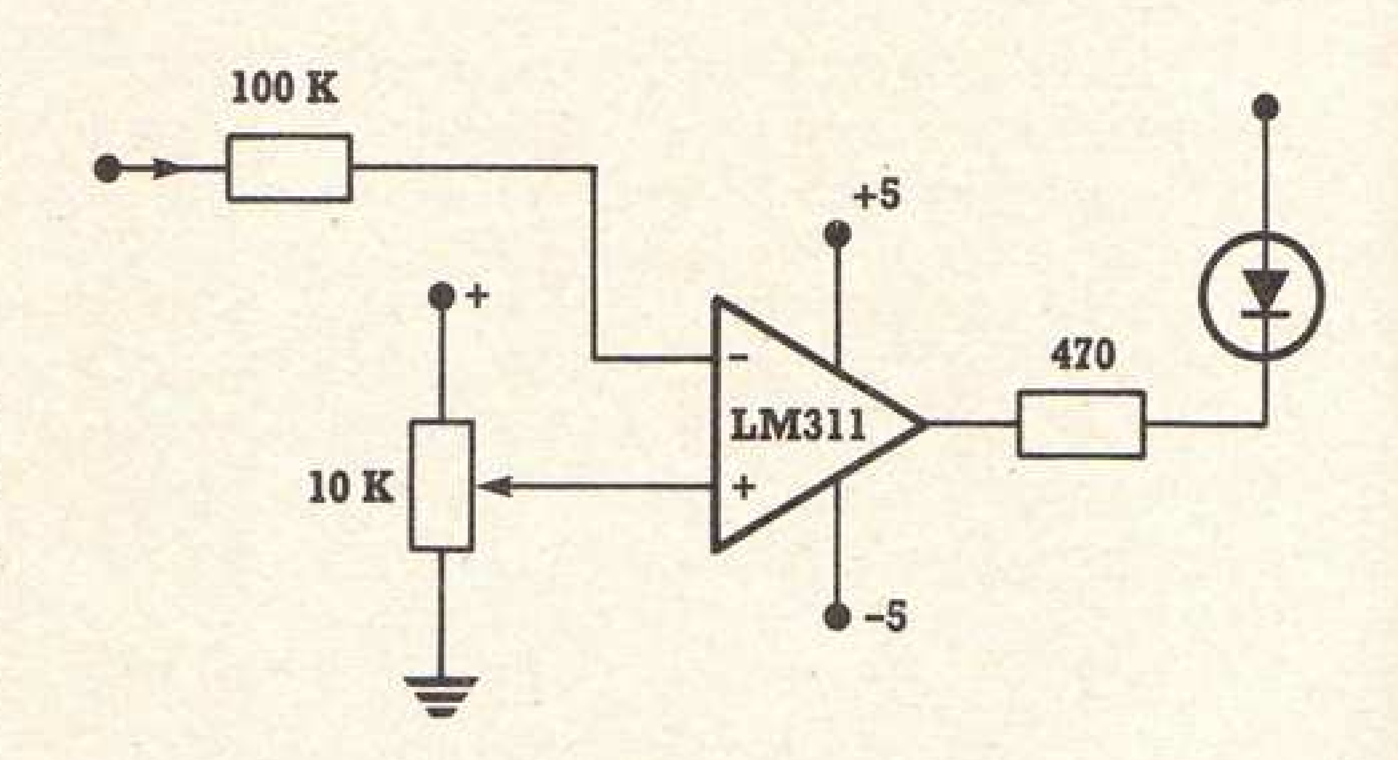


Fig. 9. Sonda de nivel variable.

activarse por la señal de entrada queda encendido hasta que se apague manualmente o por una entrada adicional.

Es muy conveniente montarse al menos una sonda de niveles y tenerla preparada, con alimentación propia o conectada a la del sistema a través de pinzas, para poder determinar estados de manera rápida. Conviene también que las sondas que montemos tengan circuito de protección de entrada, para que no

sólo no afecten al circuito a medir, sino que tampoco se deterioren al usarlas si es que medimos en un circuito de señal fuera del margen de diseño. La protección no siempre es fácil de añadir si ha de ser el circuito a prueba de locos, pero por lo menos hemos de poder admitir un margen ± 100 % sin deterioro permanente.

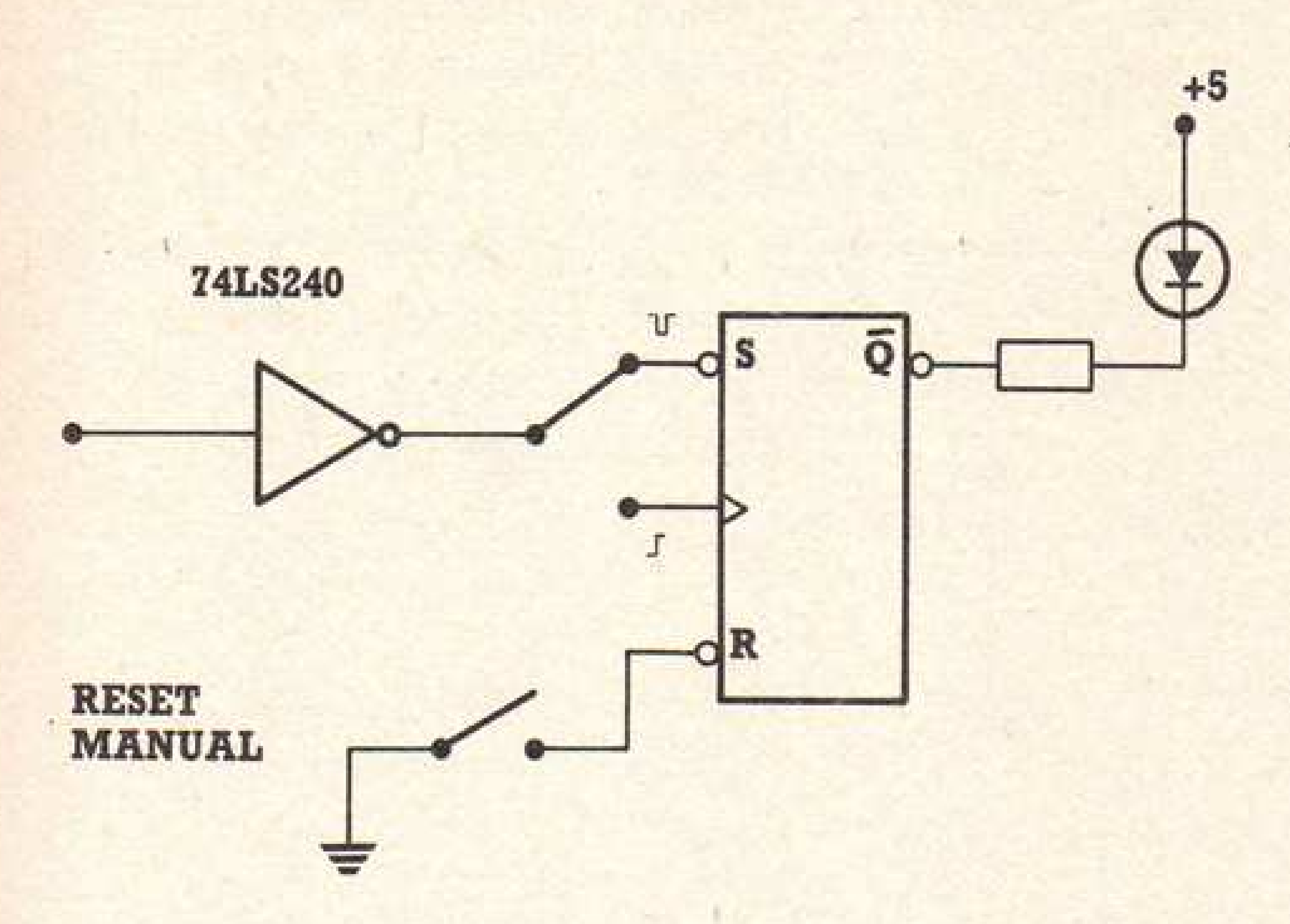


Fig. 10. Sonda con retención.

Generador de señales

Es muy conveniente disponer de un generador de señales, compatible con nuestro sistema, para poder activar algunos de los dispositivos y comprobarlos antes de conectarlos al OP. El más elemental consta de un oscilador, cuya frecuencia y ancho es modificable mediante potenciómetros. Dispuesto en una caja junto con las sondas, será de gran ayuda para la verificación de nuestros proyectos.

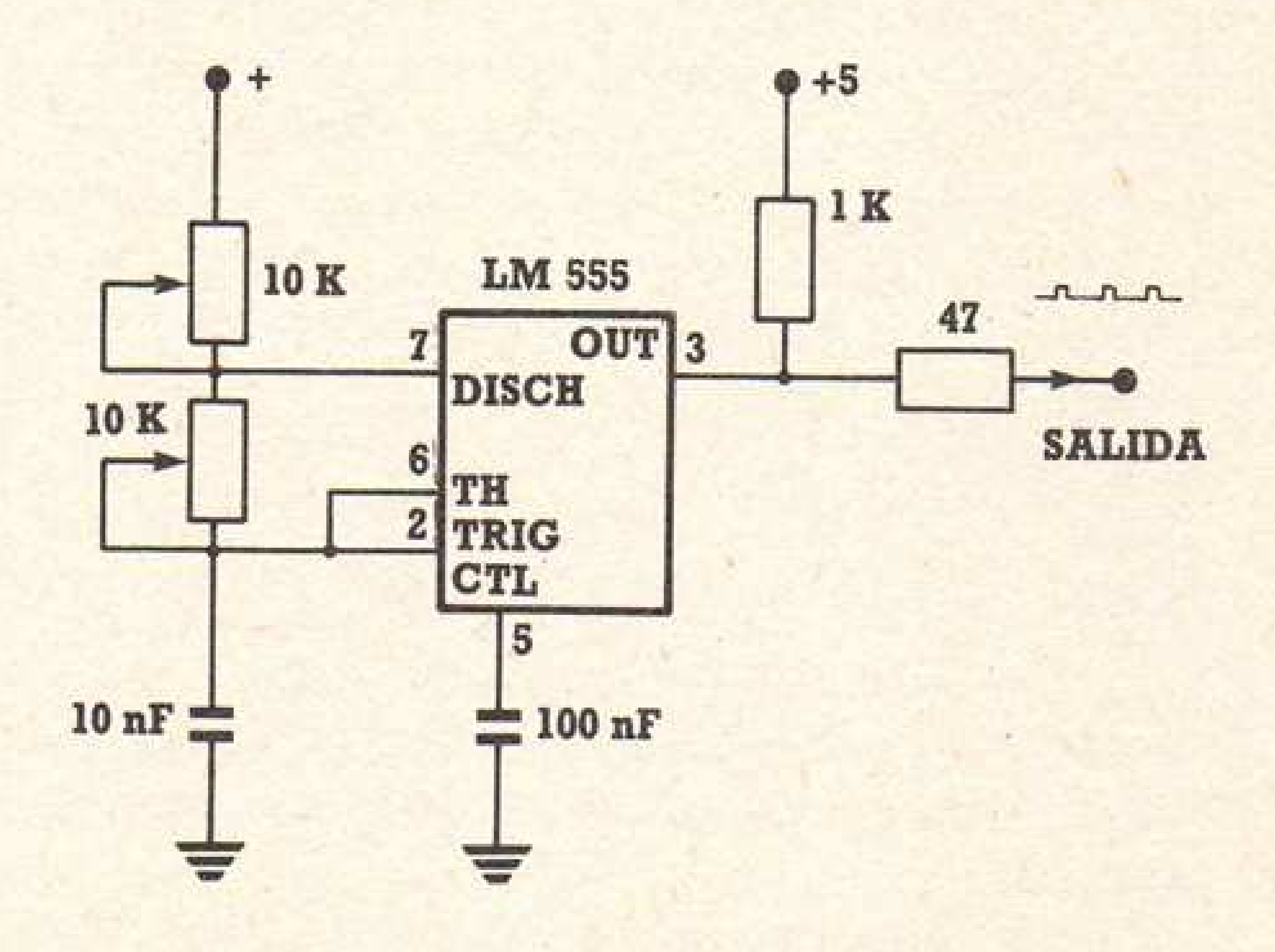


Fig. 11. Generador de señales TTL.

Fuentes de alimentación

Para poder trabajar con comodidad en el diseño y reparación de ordenadores personales y sus equipos periféricos es necesario disponer de una fuente de alimentación de prestaciones holgadas. Para los OP contemplados en nuestros diseños es necesario la tensión de 5 voltios, con una intensidad superior a 1 amperio. Sería recomendable incluso 2 amperios o más. Para algunos de los experimentos son además necesarias las tensiones de +12 y -12 voltios, por ejemplo, para interfaces de comunicaciones y analógicas. Además, para poder realizar un grabador de EPROMS es conveniente disponer de una tensión variable. Tanto esta última fuente como las de 12 voltios pueden ser de bajo consumo, por lo que podríamos sacarlas de un transformador diferente e incluso sería recomendable su montaje sólo para aquel que esté interesado en esos proyectos.

Fuente de alimentación de +5 voltios:

Proponemos una fuente de tipo convencional, con regulador lineal. En la actualidad en casi todos los OP la fuente utilizada es de tipo conmutado por su mayor rendimiento, pero aunque se dispone de circuitos para poder montar una equivalente, el problema de los transformadores de alta frecuencia lo hace poco aconsejable para un primer diseño. Nuestra fuente va a consumir unos pocos más kilovatios-hora y vamos a caldear un poquito más el ambiente.

Partimos de un transformador que convierta la tensión de red, normalmente 220 voltios de alterna, en una tensión más baja. Con un transformador de 220-6,3 @ 1 amperio lo normal es que al rectificar y filtrar con condensador resulte un tensión de pico de unos 9 voltios. Pero el rizado puede ser de más de 2 voltios, por lo que a plena carga podría aparecer rizado en la salida del regulador. Es recomendable utilizar un transformador con toma media de 7,5 voltios nominales.

El condensador de filtro se calcula para que a plena carga el rizado no haga caer la tensión a menos de 8 voltios, antes del regulador. El regulador ideal sería un 78H05, que puede dar hasta 5 amperios, si el radiador lo refrigera suficientemente. Con el 7805 podremos sacar hasta 1 amperio con un radiador suficientemente grande. Pueden montarse mejoras mediante un transistor que suministre la corriente necesaria a la carga y no directamente el regulador.

A la salida del regulador colocaremos los condensadores indicados para evitar oscilaciones, que con toda seguridad se producirían si alimentamos una tarjeta con cables largos.

La fuente podemos montarla en una caja con bornas para conexión a los equipos externos. Los indicadores LED, fusible, llave de conmutación y las letras en la carátula son recomendables para hacerla más operativa.

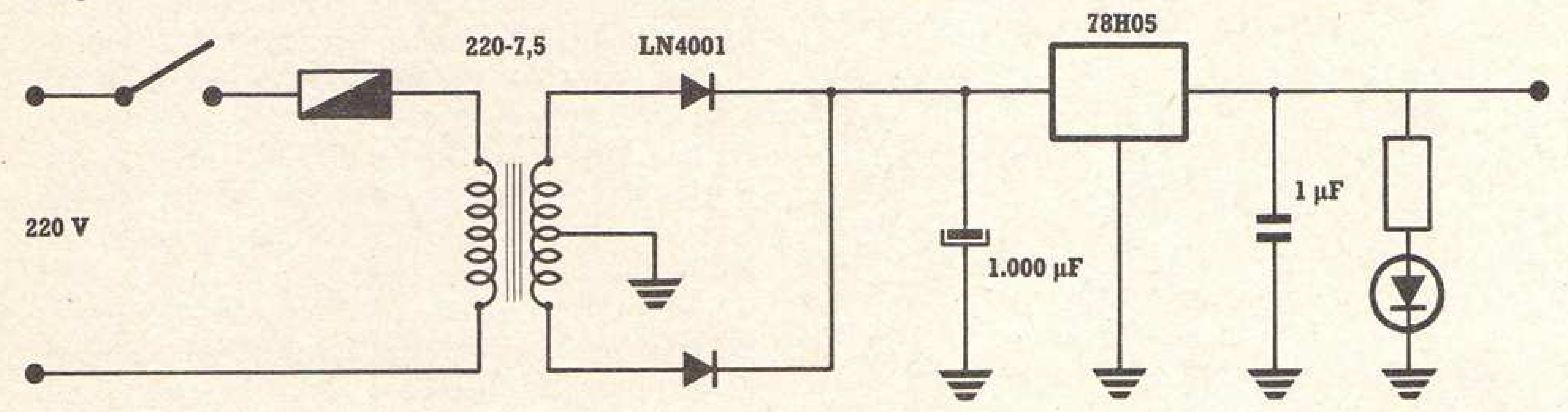


Fig. 12. Fuente de alimentación de 5 voltios.

Analizador lógico y osciloscopio

Se ponen juntos estos dos equipos porque son igualmente deseables para un buen taller profesional de diseño, pero sólo un sueño para el aficionado. Sin embargo, veremos, a lo largo de los diseños que se propondrán en lo sucesivo, cómo se pueden construir equipos similares, que aunque de prestaciones limitadas pueden aumentar nuestras posibilidades de observación del mundo que nos rodea. Al fin y al cabo los osciloscopios y analizadores actuales llevan en su corazón un micro de características muy similares a las de nuestro OP y, por tanto, si le dotamos de "ojos y oídos" suficientemente rápidos, podremos hacerle que vea y oiga igual que los equipos reales.

Organización

Es importante incluir este apartado para mostrar cómo puede influir en el rendimiento de nuestro tiempo libre la forma de cómo nos lo montemos.

Mesa de trabajo

Aunque pueda parecer chocante, es conveniente recordar que hay que contar con iluminación adecuada para que los resultados sean buenos. Una lámpara de brazo variable es lo más apropiado para acercarla donde convenga.

En primer lugar, el espacio de trabajo ha de permitirnos acceder a todos los elementos necesarios sin muchos desplazamientos y casi al primer toque, pues si no es así pronto nos cansaremos y nos dedicaremos a otra cosa. Hemos de disponer de un espacio donde quepan:

- El OP, su pantalla y teclado.
- La tarjeta de pruebas.
- La fuente de alimentación.
- El polímetro.
- El soldador y su base de soporte.
- La caja de componentes.
- Las herramientas.
- Los planos y documentación del experimento en estudio.

Puede parecer que es mucho el espacio necesario y más de uno podrá decir que lo hace con menos, esto que proponemos es lo ideal.

Caja de componentes

Los componentes propios para los montajes relacionados con nuestras actividades es conveniente tenerlos ordenados y en estado de revista, para no perder mucho tiempo en su búsqueda. Por eso distinguimos la caja de componentes ordenada de lo que viene después. Ya que son componentes generalmente caros, compraremos solamente los necesarios para cada diseño, pero es casi seguro que en poco tiempo tendremos cosas que ordenar. Además, para experimentar hay un mínimo de componentes que es necesario tener, si queremos hacer algo más que montar un kit ya diseñado. Las secciones apropiadas pueden ser:

- Circuitos integrados. Ordenados por código. Como normalmente no tendremos muchos, podrían agruparse los de números próximos.
- Resistencias, potenciómetros. Agrupadas por rangos.

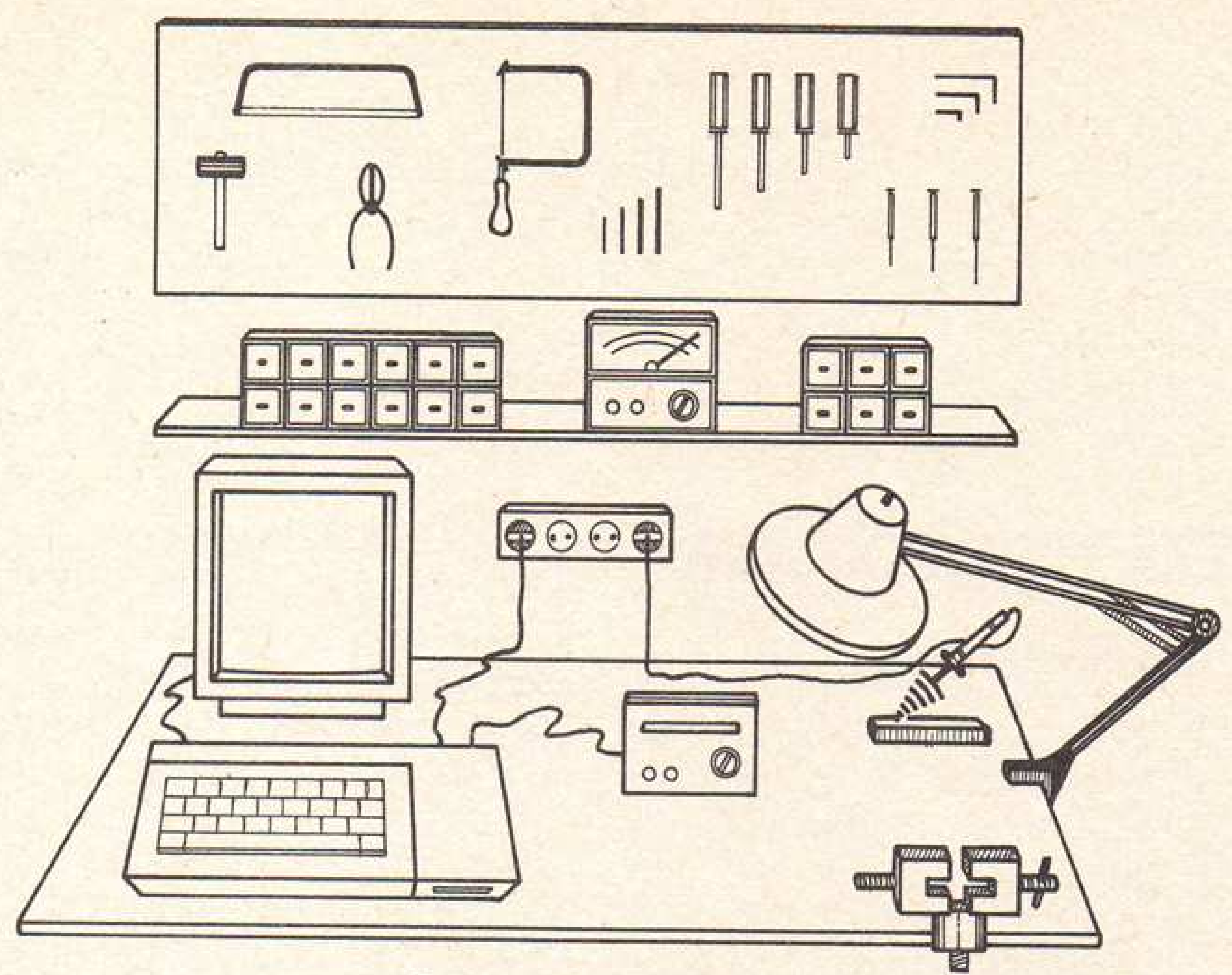


Fig. 13. Mesa de trabajo.

- Condensadores. Agrupados por rangos.
- Transistores, diodos, LEDS, etc.
- Placas de circuito impreso de tiras.
- Conectores y terminales de los apropiados para nuestro OP.
- Conmutadores, pulsadores, interruptores.
- Cables. Agrupados en mazos para su rápida disposición.
- Zócalos de circuitos.
- Tornillos, tuercas y arandelas.

En la mayoría de los casos, conviene preparar un contenedor para cada proyecto donde reservamos todos los componentes relacionados con él, para tenerlos a mano en el momento de montarlos. Una vez finalizado el trabajo podemos retornar los sobrantes a su casilla original.

Cajón de sastre

Es el lugar adonde van a parar todos los componentes perdidos o que por su pequeño número no merece la pena tener un lugar permanente para su colocación. Será el lugar adonde vayan a parar los componentes del desguace de una tarjeta vieja, los componentes sobrantes de un experimento anterior y en general los componentes de dudosa clasificación. Aunque lo ideal sería que este cajón se redujera a la mínima expresión, la realidad muestra que para cualquier experimentador activo crece sin tener límite teórico. Solamente las posibilidades físicas de espacio son las que suelen poner barreras a su crecimiento.

Libros de datos

Convendrá que con cada componente o equipo utilizado guardemos su descripción para poder conocer sus características con detalle cuando sea necesario. También convendrá que la documentación esté ordenada de acuerdo con la clasificación de componentes, por familias y tipos.

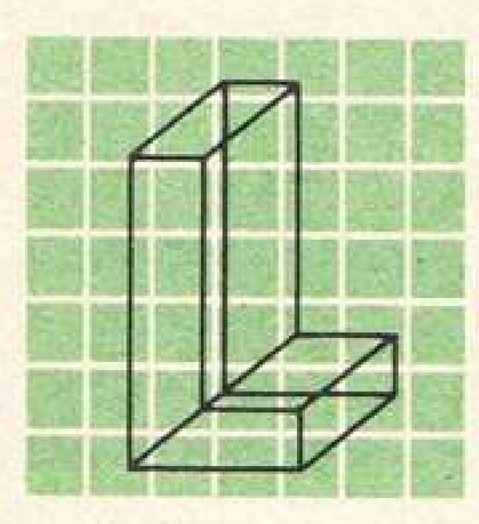
Delante de la mesa de trabajo es recomendable disponer un panel de corcho donde puedan fijarse mediante chinchetas los planos o las instrucciones que necesitemos recordar con frecuencia. Por ejemplo, el código de colores de resistencias y condensadores o la distribución de patillas de los componentes más usuales. Nos agradeceremos sobradamente el esfuerzo inicial dedicado a este tipo de información.

Botiquín

Aunque puede parecer alarmante, no está de más el recordar que una quemadura fortuita con el soldador le ocurre al más hábil, por lo que conviene no estar muy lejos de donde haya una tirita o una tintura apropiada para las quemaduras.

NATURALEZA Y TECNOLOGIA

Botánica. Las flores



AS flores son los órganos reproductores de las plantas superiores (espermafitas o fanerógamas). Una flor consta de una serie de cubiertas o "verticilos florales" que, de fuera a dentro, son: cáliz, corola, androceo y gineceo.

Los dos primeros verticilos tienen el carácter de cubiertas, reuniéndose ambos bajo el nombre de periantio; el androceo es el aparato reproductor masculino (conjunto de estambres), mientras que el gineceo constituye el aparato reproductor femenino (conjunto de pistilos).

El androceo es el conjunto de una serie de piezas denominadas estambres. Un estambre consta de un filamento alargado y una antera, cuya parte superior está formada por un par de tecas, y cada teca contiene dos sacos polínicos en los que se encierran los granos de polen (gametos masculinos).

El gineceo tiene forma de botella, con un vientre ensanchado (ovario), un cuello más o menos alargado (estilo), y una parte superior ensanchada y viscosa (estigma); el estigma recibirá y fijará sobre él el grano de polen (polinización).

La polinización consiste en el transporte de polen desde las anteras hasta el estigma. Puede verificarse por diversos medios, por ejemplo, por el viento (polinización anemógama) o por medio de los insectos (polinización entomógama). Con el fin de atraer a los insectos, la corola de muchas flores ofrece colores muy vivos. Cuando el insecto los localiza, acude a libar los jugos azucarados contenidos en los nectarios de la flor; sobre ella se impregna de polen, que será transportado por el animal hasta otra flor, teniendo lugar de esta forma una polinización cruzada.

Una segunda fase en la reproducción se da con la fecundación. Una vez fijado en el estigma el grano de polen comienza a prolongarse para formar un sifón o tubo polínico que descenderá por el estilo hasta llegar al interior de la cavidad del ovario.

Por el tubo polínico descienden los dos tubos del grano de polen: uno de ellos fecundará a la oosfera para dar lugar al embrión de la nueva planta; el segundo se unirá con el llamado núcleo secundario para dar un tejido de reserva, el albumen.

El ovario fecundado sufrirá posteriormente una serie de transformaciones (maduración), que darán lugar al fruto. El óvulo fecundado, transformado y maduro, dará por su parte la semilla. Cuando el ovario y el óvulo han sido fecundados y alcanzan su estado de maduración, el periantio se marchita y se desprende. El fruto se abre y deja en libertad a las semillas, con capacidad de germinar y dar lugar a una nueva planta.

Sobre el programa

El programa consiste en preguntas sobre algunas nociones de las flores como las que os acabamos de contar.

Hemos utilizado un gráfico en el que puedes ver las distintas partes de la flor; te

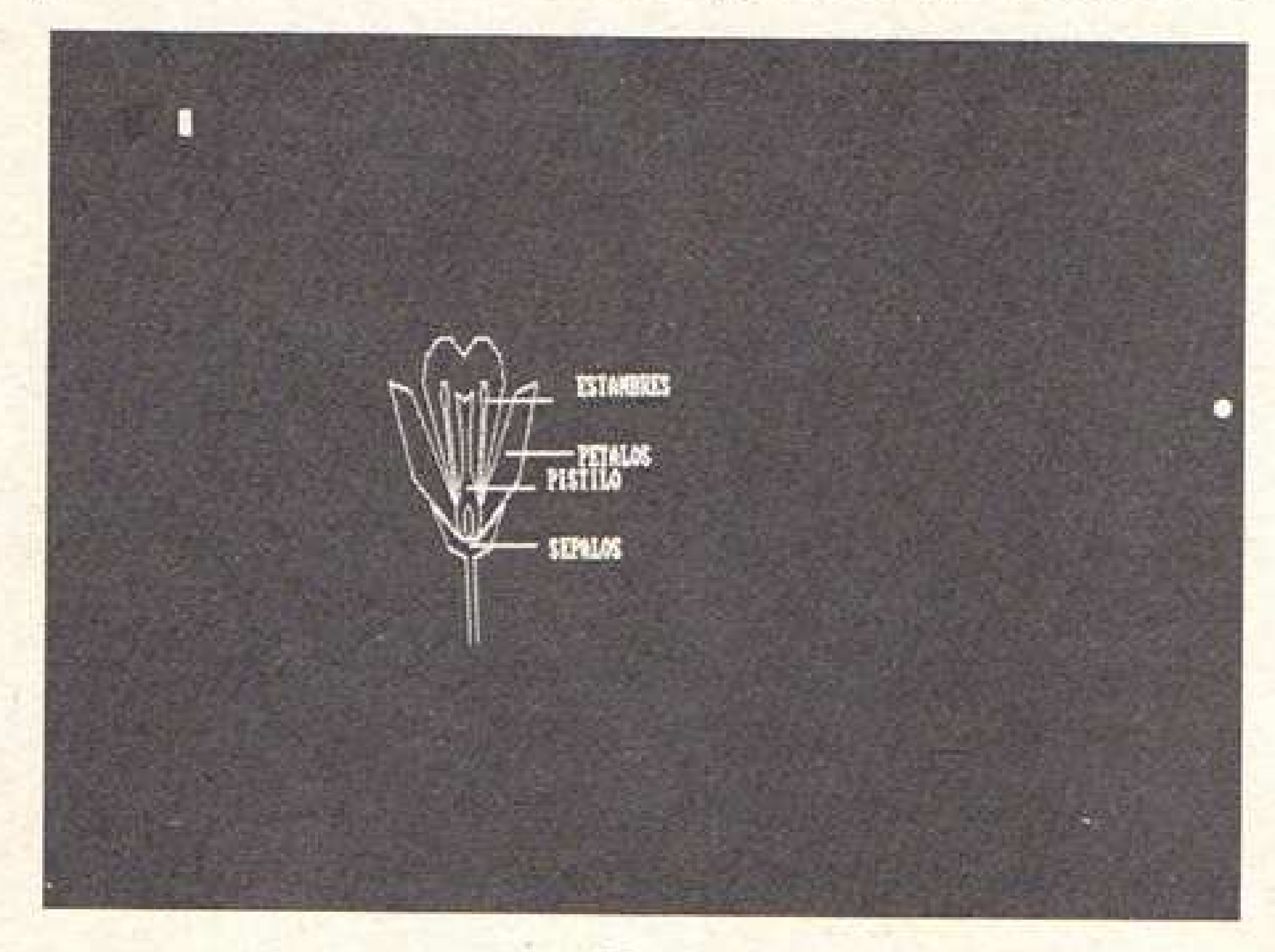


Fig. 1.

servirá para utilizar la memoria visual y responder correctamente a las preguntas.

Debes responder siempre con mayúsculas.

```
10 REM ****************************
20 REM **  PROGRAMA DE BOTANICA  **
30 REM **        LA FLOR         **
40 REM ****************************
50 SCREEN 2
60 LET C=0
70 GOSUB 3000
80 FOR P=1 TO 10
90 GOSUB 1000
100 IF R$=B$ THEN LET C=C+1: GOTO 120
110 GOSUB 2000
120 NEXT P
130 PRINT "HAS TENIDO ";C;"RESPUESTAS CORRECTAS"
140 GOSUB 4000
150 END
1000 REM ***************************
1010 REM ** SUBRUTINA DE PREGUNTA **
1020 REM ***************************
1030 READ A$
1040 LOCATE 1,1: PRINT A$
1050 READ B$
1060 LOCATE 2,1
1070 PRINT "
1080 LOCATE 2,1
1090 INPUT R$
1100 RETURN
2000 REM ***********************
2010 REM ** RESPUESTA ERRONEA **
2020 REM ***********************
2030 LOCATE 2,1
2040 PRINT "LA RESPUESTA ES ";B$
2050 FOR Z=1 TO 1000:NEXT Z
2060 RETURN
3000 REM *************************
3010 REM ** SUBRUTINA DE DIBUJO **
3020 REM *************************
3030 CLS
3040 LET Y=280-100
3050 PSET (200,Y)
3060 FOR I=1 TO 9
3070 READ A$,B: LET B=B*(-1): LET B$=STR$(B)
3080 LET DIB$="M"+A$+","+B$
3090 DRAW DIB$
3100 NEXT I
3110 LET Y=280-136
3120 PSET (197,Y)
3130 FOR I =1 TO 54
3140 READ A$,B: LET B=B*(-1): LET B$=STR$(B)
3150 LET DIB$="M"+A$+","+B$
3160 DRAW DIB$
3170 NEXT I
3180 RETURN
4000 REM ****************************
4010 REM ** NOMBRES PARA EL DIBUJO **
4020 REM ****************************
4030 FOR I=1 TO 4
4040 READ A,B
4050 PSET(A,280-B)
4060 DRAW "M+50,+0"
4070 READ Q$
4080 READ A,B
4090 LOCATE A,B
4100 PRINT Q$
4110 NEXT I
4120 RETURN
5000 DATA +0,+30,-12,+3,-12,+15,+21,-12
5010 DATA +12,+0,+21,+12,-12,-15,-12,-3
5020 DATA +0,-30
5030 DATA -3,+6,+0,+6,+6,+12,+0,+12
5040 DATA -3,+15,+6,-3,+6,+3,-3,-15
5050 DATA +0,-12,+6,-12,+0,-6,-3,-6
5060 DATA -6,+0,-3,+3,+0,+6,+3,+3,+3,-3
5070 DATA +0,-6,-3,-3,-6,+0
5080 DATA -33,+21,-15,+33
5090 DATA +15,-3,+30,-39,-6,+33,-2,+3
5100 DATA +1,+6,+3,+0,+1,-6,-1,-3,+6,-30
5110 DATA -2,-3,-21,+42,+3,+9,+9,+6
5120 DATA +9,+0,+9,-6,+9,+6,+9,+0,+9,-6,+3,-9,-21,-42,-2,+3
5130 DATA +6,+30,-1,+3,+1,+6,+3,+0,+1,-6
```

```
5140 DATA -2,-3,-6,-33
5150 DATA +30,+39
5160 DATA +15,+3,-15,-33,-33,-21
6000 DATA "LAS HOJAS DE COLOR VERDE DE LA BASE DE LA FLOR SON LOS "
6010 DATA "SEPALOS"
6020 DATA "EL CONJUNTO DE LOS SEPALOS FORMA EL                      "
6030 DATA "CALIZ"
6040 DATA "LAS HOJAS COLOREADAS EN LA PARTE SUPERIOR SON LOS       "
6050 DATA "PETALOS"
6060 DATA "EL CONJUNTO DE LOS PETALOS FORMA LA                     "
6070 DATA "COROLA"
6080 DATA "LOS GRANOS DE POLEN SE ENCUENTRAN EN LA                 "
6090 DATA "ANTERA"
6100 DATA "LA PARTE QUE TIENE FORMA DE BOTELLA ES EL               "
6110 DATA "PISTILO"
6120 DATA "LA REPRODUCCION EMPIEZA CON LA                          "
6130 DATA "POLINIZACION"
6140 DATA "EL ANDROCEO DE LA FLOR LO FORMAN LOS                    "
6150 DATA "ESTAMBRES"
6160 DATA "CADA ANTERA ESTA FORMADA POR DOS                        "
6170 DATA "TECAS"
6180 DATA "LAS PARTES DEL PISTILO SON: OVARIO, ESTILO Y ...        "
6190 DATA "ESTIGMA"
7000 DATA 203,134,"SEPALOS",19,34
7010 DATA 233,166,"PETALOS",15,37
7020 DATA 203,154,"PISTILO",16,34
7030 DATA 218,184,"ESTAMBRES",12,37
```

Modificaciones para otros equipos

Variaciones para el ordenador AMSTRAD.

Haz las siguientes variaciones en el programa:

```
50 MODE 2            3110 no se pone
1060 LOCATE 1,2      3120 PLOT 197,136
1080 LOCATE 1,2      3140 READ A,B
1140 LOCATE 1,2      3150 no se pone
3040 no se pone      3160 DRAWR A,B
3050 PLOT 200,100    4050 PLOT A,B
3070 READ A,B        4060 DRAWR 50,0
3080 no se pone      4090 LOCATE B,A
3090 DRAWR A,B
```

Variaciones para el ordenador ZX-SPECTRUM.

```
50 no se pone
150 GOTO 9999
1040 PRINT AT 1,1; A$
1060 PRINT AT 2,1;" "
1070 no se pone
1080 no se pone
2030 PRINT AT 2,1; "LA RESPUESTA ES ";B$
2040 no se pone
2050 FOR Z=1 TO 500: NEXT Z    3140 READ A,B
3040 no se pone                3150 no se pone
3050 PLOT 200,100              3160 DRAW A,B
3070 READ A,B                  4050 PLOT A,B
3080 no se pone                4060 DRAW 50,0
3090 DRAW A,B                  4090 PRINT AT A,B; Q$
3110 no se pone                4100 no se pone
3120 PLOT 197,136
```

Variaciones para MSX

```
50 SCREEN 1
```

Variaciones para COMMODORE

```
50 PRINT CHR$(147)
70 no se pone
140 no se pone
1040 PRINT A$
1060 no se pone
1070 PRINT
1080 no se pone
2030 no se pone
```

Por último, queda recordarte que puedes variar las preguntas que se formulan cuando éstas ya te resulten fáciles. Para ello no tienes más que situarlas en los DATAs correspondientes que se encuentran a partir de la línea 6000.

MATEMATICAS

Función lineal

En esta ocasión, el ordenador va a ayudarnos a conocer las funciones lineales.

Una de las capacidades más sofisticadas y útiles del ordenador es la posibilidad de representación de funciones.

Hemos elegido la función lineal por ser ésta la más fácil de comprender. Permite por su sencillez adquirir una agilidad en la representación y entender su significado.

En general, la forma de la ecuación lineal es:

$$Y = aX + b$$

a- Es la pendiente de la recta. Es decir,

una variación de "a" cambiará la inclinación de nuestra recta.

b- Es el valor de la ordenada en el origen. Es decir, el valor que tiene el eje de las ordenadas o eje de la Y cuando el de X vale cero. Si lo haces aplicándolo a la ecuación verás cómo al ser X = 0 el término (aX) vale cero, con lo que te queda Y = b que, como ya hemos dicho, es el valor de la ordenada en el origen.

Un caso particular de estas funciones se da cuando la ordenada en el origen vale cero, o lo que es lo mismo si b = 0 tenemos que Y = 0, la recta pasará por el mismo origen de coordenadas (0,0) y en este caso la función se llama AFIN.

$$Y = aX$$

Otro caso se produce cuando a=0. Entonces la ecuación de la recta queda:

$$Y = b$$

que es la ecuación de una recta paralela al eje X en el punto b.

Siguiendo con estos casos, si a = 0 y b = 0 la ecuación queda:

$$Y = 0$$

que coincide con la ecuación del eje de abscisas o eje X.

Programa

Pasamos ahora al estudio del programa. Las partes de que consta son:

— Asignación de los datos. Donde se piden los parámetros a y b, se asignan otros que son necesarios para la representación y se calculan los distintos valores de la función haciendo una tabla.

— Determinación de la escala con que se va a representar en función de la magnitud de los valores calculados en la primera parte.

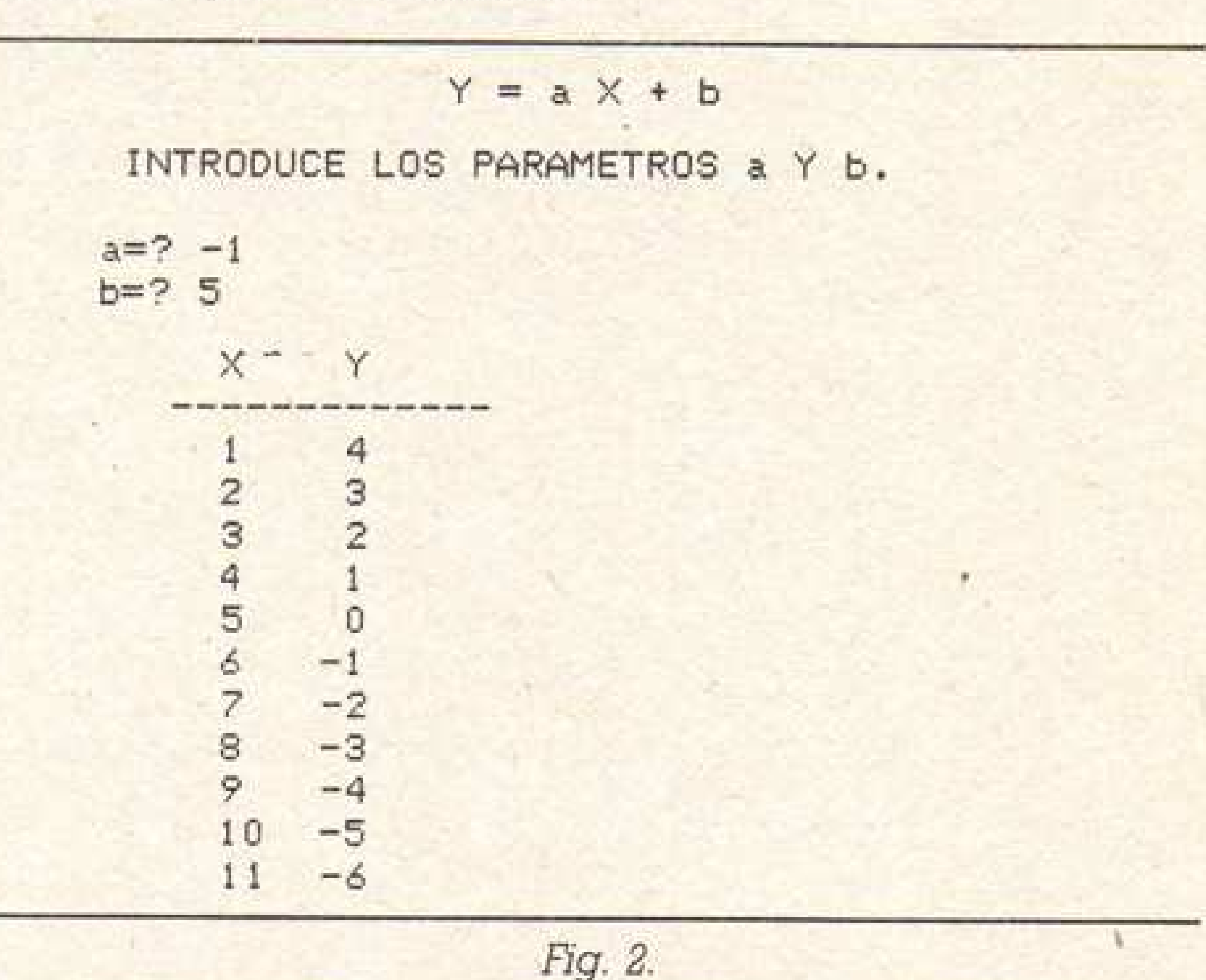
— Situación del origen.

— Representación del eje X.

— Se dibuja el eje Y al mismo tiempo que se representan con "*" los valores de la función.

— Se vuelve a dibujar el eje X para terminar la gráfica.

— Se dan el primer y último valor de la abscisa y la ordenada.



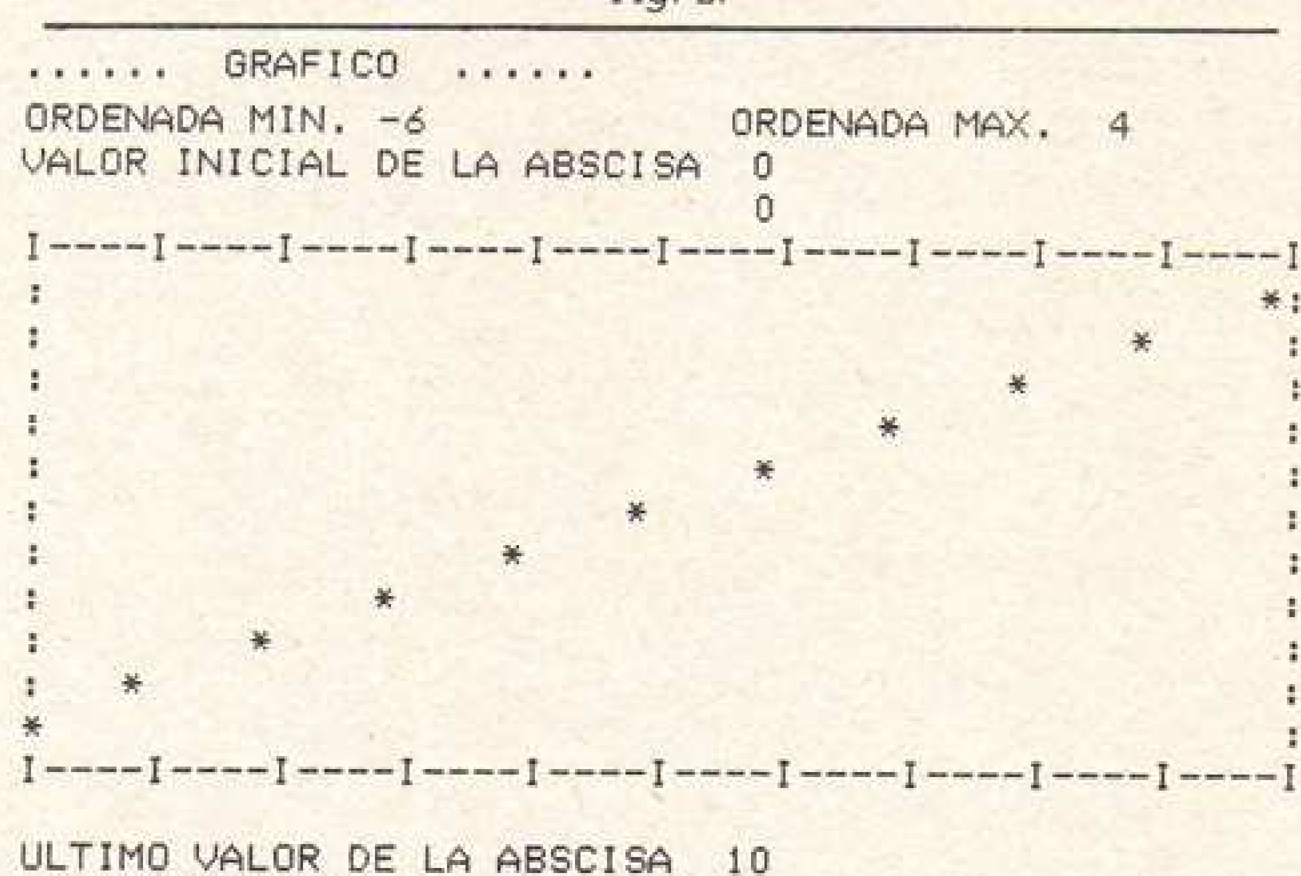
```
                 Y = a X + b

   INTRODUCE LOS PARAMETROS a Y b.

a=? -1
b=? 5

      X    Y
   -------------
      1    4
      2    3
      3    2
      4    1
      5    0
      6   -1
      7   -2
      8   -3
      9   -4
     10   -5
     11   -6
```

Fig. 2.



```
...... GRAFICO ......
ORDENADA MIN. -6              ORDENADA MAX.  4
VALOR INICIAL DE LA ABSCISA  0
                             0
I----I----I----I----I----I----I----I----I----I----I
:                                                *:
:                                           *     :
:                                      *          :
:                                 *               :
:                            *                    :
:                       *                         :
:                  *                              :
:             *                                   :
:        *                                        :
:   *                                             :
*                                                 :
I----I----I----I----I----I----I----I----I----I----I

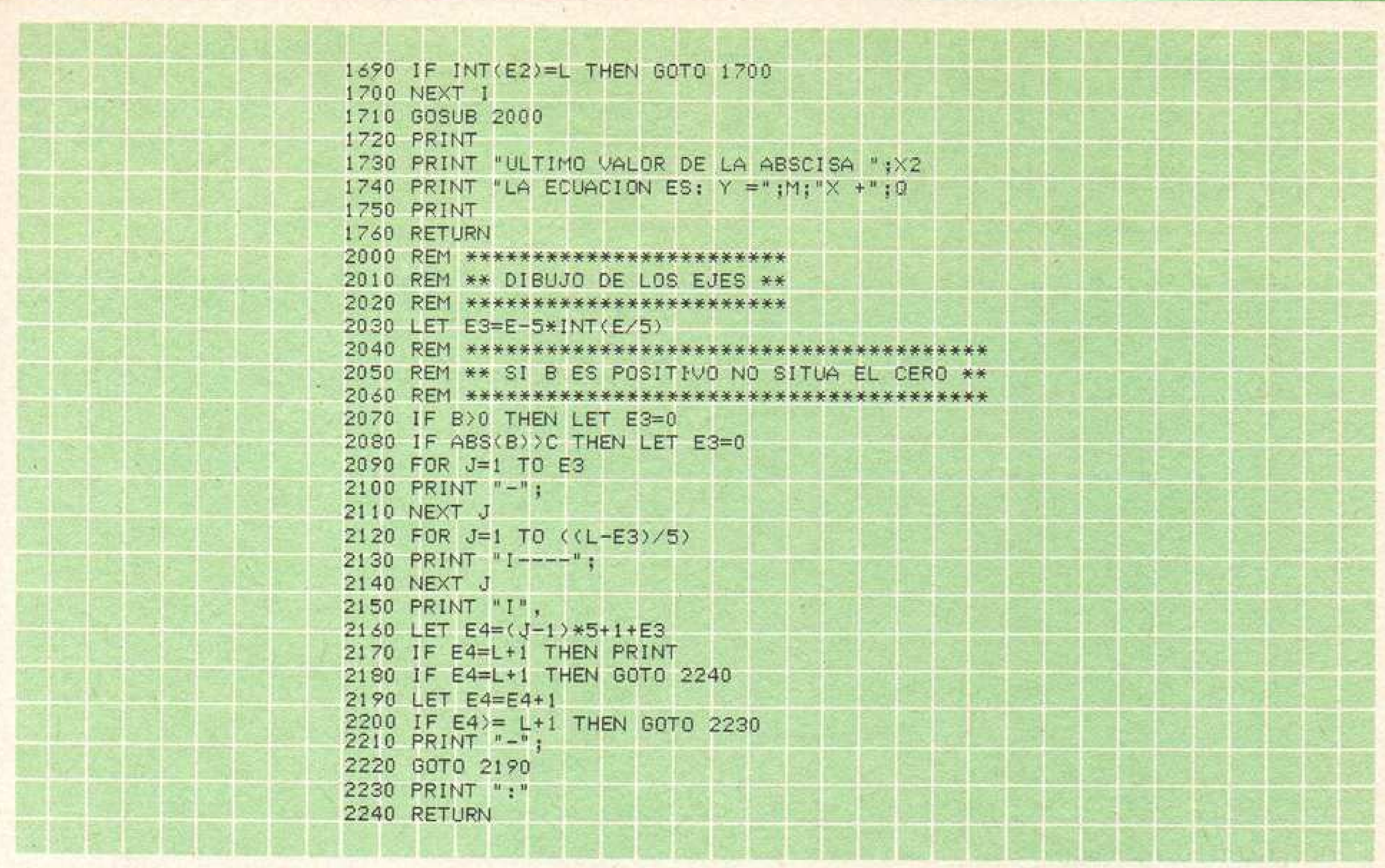
ULTIMO VALOR DE LA ABSCISA  10
LA ECUACION ES: Y =-1 X + 5
```

Fig. 3.



```
10 CLS
20 REM ****************************************
30 REM **   ESTE PROGRAMA HACE UN GRAFICO DE   **
40 REM **     VALORES IGUALMENTE ESPACIADOS    **
50 REM **        DE UNA FUNCION LINEAL         **
60 REM ****************************************
70 PRINT " LA ECUACION DE LA FUNCION LINEAL ES:": PRINT
80 PRINT "                Y = a X + b ": PRINT
90 PRINT " INTRODUCE LOS PARAMETROS a Y b."
100 PRINT
110 REM **********************
120 REM ** INTRODUCIR DATOS **
130 REM **********************
140 INPUT "a=";M: INPUT "b=";O
150 PRINT
160 REM **  LONGITUD FIJA **
170 LET L=50
180 REM ** PRIMERA COORDENADA FIJA **
190 LET X1=0
200 REM **  LA ULTIMA COORDENADA FIJA **
```

```
210 LET X2=10
220 REM ** NUMERO DE PUNTOS A DIBUJAR FIJO **
230 LET N=11
240 DIM D(N+1)
250 PRINT TAB(6) "X";TAB(11) "Y"
260 PRINT "    --------------"
270 FOR I=1 TO N
280 LET D(I)=M*I+Q
290 PRINT TAB(5);I; TAB(10);D(I)
300 NEXT I
310 FOR Z=1 TO 5000: NEXT Z
320 GOSUB 1000
330 END
1000 REM ***************************
1010 REM ** SUBRUTINA DE GRAFICOS **
1020 REM ***************************
1030 LET B=100000000#
1040 REM ****************************************
1050 REM ** BUSCA EL VALOR MINIMO DE LOS DATOS **
1060 REM ****************************************
1070 FOR I=1 TO N
1080 IF B>D(I) THEN LET B=D(I)
1090 NEXT I
1100 REM ****************************************
1110 REM ** RESTAR A TODOS LOS DATOS EL MINIMO **
1120 REM ****************************************
1130 FOR I=1 TO N
1140 LET D(I)=D(I)-B
1150 NEXT I
1160 REM ********************************
1170 REM ** BUSCA LA MAXIMA DIFERENCIA **
1180 REM ********************************
1190 LET C=0
1200 FOR I=1 TO N
1210 IF C<D(I) THEN LET C=D(I)
1220 NEXT I
1230 REM **************************
1240 REM ** DETERMINAR LA ESCALA **
1250 REM **************************
1260 LET A=L/C
1270 REM ***********************************
1280 REM ** BUSCAR LA POSICION DEL ORIGEN **
1290 REM ***********************************
1300 LET O=A*ABS(B)
1310 PRINT
1320 PRINT "......  GRAFICO  ......"
1330 PRINT
1340 PRINT "ORDENADA MIN. ";B,"ORDENADA MAX. ";C+B
1350 PRINT "VALOR INICIAL DE LA ABSCISA ";X1
1360 REM ************************************************
1370 REM ** SI B ES POSITIVO NO HAY QUE SITUAR EL CERO **
1380 REM ************************************************
1390 IF B>0 THEN GOTO 1480
1400 REM ***************************************************************
1410 REM ** SI TODOS LOS DATOS SON NEGATIVOS TAMPOCO SE SITUA EL CERO **
1420 REM ***************************************************************
1430 IF ABS(B)>C THEN GOTO 1480
1440 REM ************************
1450 REM ** SITUACION DEL CERO **
1460 REM ************************
1470 PRINT TAB(O);"0"
1480 GOSUB 2000
1490 FOR I=1 TO N
1500 IF INT(.2*L/N)<1 THEN GOTO 1600
1510 REM ****************************************************
1520 REM ** INTRODUCIR LA MARCA DE ESPACIOS DE LA ORDENADA **
1530 REM ****************************************************
1540 FOR K=1 TO (INT(.6*L/N))
1550 PRINT ":" TAB(L+1) ":"
1560 NEXT K
1570 REM ***************************
1580 REM ** POSICION DE LOS DATOS **
1590 REM ***************************
1600 LET E2=A*D(I)
1610 REM **************************************
1620 REM ** LOCALIZAR LOS PUNTOS CON "*"     **
1630 REM **************************************
1640 IF E2>=1 THEN GOTO 1670
1650 PRINT "*";TAB(L+1) ":"
1660 GOTO 1700
1670 PRINT ":";
1680 PRINT TAB(E2) "*";TAB(L+1) ":"
```



```
1690 IF INT(E2)=L THEN GOTO 1700
1700 NEXT I
1710 GOSUB 2000
1720 PRINT
1730 PRINT "ULTIMO VALOR DE LA ABSCISA ";X2
1740 PRINT "LA ECUACION ES: Y =";M;"X +";Q
1750 PRINT
1760 RETURN
2000 REM ************************
2010 REM ** DIBUJO DE LOS EJES **
2020 REM ************************
2030 LET E3=E-5*INT(E/5)
2040 REM **************************************
2050 REM ** SI B ES POSITIVO NO SITUA EL CERO **
2060 REM **************************************
2070 IF B>0 THEN LET E3=0
2080 IF ABS(B)>C THEN LET E3=0
2090 FOR J=1 TO E3
2100 PRINT "-";
2110 NEXT J
2120 FOR J=1 TO ((L-E3)/5)
2130 PRINT "I----";
2140 NEXT J
2150 PRINT "I",
2160 LET E4=(J-1)*5+1+E3
2170 IF E4=L+1 THEN PRINT
2180 IF E4=L+1 THEN GOTO 2240
2190 LET E4=E4+1
2200 IF E4>= L+1 THEN GOTO 2230
2210 PRINT "-";
2220 GOTO 2190
2230 PRINT ":"
2240 RETURN
```

Para este programa se ha utilizado un BASIC muy general que te permitirá adaptar el programa a cualquier ordenador.

Debes tener en cuenta que si tu ordenador es un COMMODORE tienes que sustituir la línea 10 de la siguiente forma:

10 PRINT CHR$(147)

Y si tienes un SPECTRUM cambias la línea 330:

330 GOTO 9999

Por último, te aconsejamos que guardes este programa de forma segura, ya que la subrutina de representación gráfica la vamos a utilizar en sucesivas ocasiones.

SOCIEDAD

Literatura. La generación del 98

El nombre de Generación del 98 suele darse a un grupo de escritores que se dieron a conocer a principios del siglo XX y en los que influyó el desastre colonial español de 1898.

Para los historiadores una generación es un conjunto de personas que tiene aproximadamente la misma edad y que, por tanto, comparten problemas e inquietudes, y se ven obligados a reaccionar ante los mismos acontecimientos. Sin embargo, no basta esto, son necesarios además, una formación intelectual semejante, algún tipo de contacto entre ellos, un acontecimiento nacional que aúne sus voluntades (en nuestro caso el mencionado desastre colonial español) y rasgos comunes en el estilo por los que se opongan a la generación anterior.

La generación literaria del 98 la forman: Ganivet (como precursor), *Azorín,* Baroja y Maeztu (como grupo inicial), Unamuno, Antonio Machado y Valle-Inclán, y Menéndez Pidal como científico.

Es común a todos ellos una preocupación por España, un deseo de superar la retórica hueca del siglo XIX y de aproximarse a las entrañas mismas del país, de enfocar con realismo los problemas, no para resolverlos en el campo de la acción social o política, aunque muchos de ellos hayan intervenido en política con mayor o menor fortuna, sino para despertar la conciencia nacional. Ello les lleva a la revalorización del paisaje y, especialmente, del paisaje castellano; al redescubrimiento de los primitivos (Berceo, Arcipreste de Hita), de los clásicos (Góngora), del Greco (ya valorado en Cataluña).

La renovación literaria, perseguida por todos ellos, con la máxima sinceridad y naturalidad huyendo de la retórica, produce en cada autor un estilo personalísimo inconfundible.

El programa

Consta de un test de diez preguntas. Para que vayas memorizando las respuestas correctas y perfecciones tus conocimientos, cada vez que fallas, aparece la respuesta que deberías haber dado.

```
10 REM **********************
20 REM **TEST DE LITERATURA**
30 REM **********************
40 CLS
50 COT=0
60 RANDOMIZE TIMER
70 DIM P$(10):DIM S$(30):DIM K(10)
80 REM ***********************************
90 REM **LECTURA DE PREGUNTAS Y RESPUESTAS**
100 REM ***********************************
110 FOR J=1 TO 10
120 READ P$(J)
130 NEXT J
140 FOR I=1 TO 30
150 READ S$(I)
160 NEXT I
170 FOR N=1 TO 10
180 READ K(N)
190 NEXT N
200 FOR I=1 TO 10
210 PRINT TAB(2);P$(I)
220 PRINT
230 PRINT TAB(5);S$(I*3-2)
240 PRINT
250 PRINT TAB(5);S$(I*3-1)
260 PRINT
270 PRINT TAB(5);S$(I*3)
280 PRINT
290 INPUT "RESPUESTA";T
300 IF T=K(I) THEN GOSUB 1000
310 IF T<>K(I) THEN GOSUB 2000
320 NEXT I
330 PRINT :PRINT :PRINT
340 PRINT TAB(12);"RESPUESTAS CORRECTAS ";COT
350 PRINT:PRINT
360 IF COT<5 THEN PRINT TAB(12);"TIENES QUE ESTUDIAR MAS"
370 IF COT>8 THEN PRINT TAB(12);"¡¡¡FELICIDADES!!! TIENES UN SOBRESALIENTE"
380 END
1000 REM **********************************
1010 REM ** SUBRUTINA RESPUESTA CORRECTA **
1020 REM **********************************
1030 PRINT "                        ¡¡¡CORRECTO!!!"
1040 LET COT=COT+1
1050 FOR A=0 TO 1000:NEXT A
1060 CLS
1070 RETURN
2000 REM ************************************
2010 REM ** SUBRUTINA RESPUESTA INCORRECTA **
2020 REM ************************************
2030 PRINT "                          INCORRECTO"
2040 PRINT
2050 PRINT "      LA REPUESTA CORRECTA ES  ";S$(I*3+K(I)-3)
2060 FOR A=0 TO 1000:NEXT A
2070 CLS
2080 RETURN
3000 DATA"DE QUIEN ES LA OBRA ESPERPENTOS","QUIEN NO ES DE ESTA EPOCA","QUIEN NA
CIO ANTES","QUIEN HABIA PERDIDO EL BRAZO IZQUIERDO"
3010 DATA "CUAL DE ESTAS OBRAS NO ES DE ANTONIO MACHADO","DE QUIEN ES LA OBRA 'S
ONATAS'","LA GENERACION DEL 98 SE DA..."
3020 DATA "DE QUIEN ES 'ANOCHE CUANDO DORMIA, SOÑE, BENDITA ILUSION, QUE UNA FON
TANA FLUIA, DENTRO DE MI CORAZON'"," CUAL DE ESTAS OBRAS ESTA ESCRITA EN PROSA",
"PIO BAROJA NACIO EN..."
3030 DATA "1 VALLE-INCLAN","2 BAROJA","3 AZORIN"
3040 DATA "1 GARCILASO","2 BAROJA","3 MAEZTU","1 MACHADO","2 VALLE-INCLAN","3 UN
AMUNO","1 AZORIN","2 VALLE-INCLAN","3 UNAMUNO"
3050 DATA "1 SOLEDADES","2 CAMPOS DE CASTILLA","3 SONATAS","1 VALLE-INCLAN","2 M
AEZTU","3 BAROJA","1 DE 1898 A 1925","2 DE 1898 A 1927","3 DE 1900 A 1910"
3060 DATA "1 VALLE-INCLAN","2 A. MACHADO","3 UNAMUNO"
3070 DATA "1 SONATA DE OTOÑO","2 SOLEDADES","3 CAMPOS DE CASTILLA","1 BILBAO","2
 PONTEVEDRA","3 SAN SEBASTIAN"
3080 DATA 1,1,3,2,3,1,2,2,1,3
```

Variaciones para ZX-SPECTRUM	380 GOTO 9999
60 no se pone	1050 FOR A=0 TO 500: NEXT A
70 DIM P$(10,40):DIM S$(30,20):DIM K(10)	2040 FOR A=0 TO 500: NEXT A
Variaciones para COMMODORE	40 PRINT CHR$(147) 1060 PRINT CHR$(147)

PARA LOS MAS JOVENES

Operaciones con fracciones

El siguiente programa sirve para operar con fracciones. Se pueden sumar fracciones, restarlas, multiplicarlas y dividirlas. Para ello, habrá que introducir los datos de las dos fracciones con las que se quiera operar. Los datos se introducen separando numerador denominador. Se obtendrá como resultado otra fracción simplificada lo máximo posible.

```
10 REM ***PRESENTACION***
20 CLS
30 PRINT:PRINT:PRINT TAB(6);"F R A C C I O N E S"
40 PRINT :PRINT :PRINT TAB(10);"1 SUMA":PRINT
50 PRINT :PRINT :PRINT TAB(10);"2 RESTA":PRINT
60 PRINT :PRINT :PRINT TAB(10);"3 PRODUCTO":PRINT
70 PRINT :PRINT :PRINT TAB(10);"4 DIVISION":PRINT
90 INPUT "OPCION ELEGIDA";X
100 IF X=1 THEN GOTO 200
110  IF X=2 THEN GOTO 350
120 IF X=3 THEN GOTO 500
130 IF X=4 THEN GOTO 600
140 GOTO 10
150 LIST
200 REM ***SUMA DE FRACCIONES***
210 GOSUB 900
220 PRINT :PRINT :PRINT :PRINT
230 PRINT N1;"/";D1;"+";N2;"/";D2;"=";
240 LET X(1)=D1:LET X(2)=D2
250 GOSUB 1000
260 LET S=((B/D1)*N1)+((B/D2)*N2)
270 PRINT S;"/";B;
280 LET X(1)=S:LET X(2)=B
290 GOSUB 1000
300 PRINT "=";X(1)/C;"/";X(2)/C
310 GOSUB 2000
320 GOTO 10
350 REM ***RESTA DE FRACCIONES***
360 GOSUB 900
370 PRINT :PRINT :PRINT :PRINT
380 PRINT N1;"/";D1;"-";N2;"/";D2;"=";
390 LET X(1)=D1:LET X(2)=D2
400 GOSUB 1000
410 LET R=((B/D1)*N1)-((B/D2)*N2)
420 PRINT R;"/";B;
430 LET X(1)=R:LET X(2)=B
440 GOSUB 1000
450 PRINT "=";X(1)/C;"/";X(2)/C
460 GOSUB 2000
470 GOTO 10
500 REM ***MULTIPLICACION FRACCIONES***
510 GOSUB 900
520 PRINT :PRINT :PRINT :PRINT
530 PRINT N1;"/";D1;"x";N2;"/";D2;"=";
540 PRINT N1*N2;"/";D1*D2;
550 LET X(1)=N1*N2:LET X(2)=D1*D2
560 GOSUB 1000
570 PRINT "=";X(1)/C;"/";X(2)/C
580 GOSUB 2000
590 GOTO 10
600 REM ***DIVISION DE FRACCIONES***
610 GOSUB 900
620 PRINT :PRINT :PRINT :PRINT
630 PRINT N1;"/";D1;":";N2;"/";D2;"=";
640 PRINT N1*D2;"/";D1*N2;
650 LET X(1)=N1*D2:LET X(2)=D1*N2
660 GOSUB 1000
670 PRINT "=";X(1)/C;"/";X(2)/C
680 GOSUB 2000
690 GOTO 10
900 REM ***INTRODUCCION DE DATOS***
910 CLS:INPUT "NUM. FRACCION 1:";N1
920 INPUT "DENOM. FRACCION 1:";D1
930 INPUT "NUM. FRACCION 2:";N2
940 INPUT "DENOM. FRACCION 2:";D2
950 RETURN
1000 REM ***CALCULO MCM Y MCD***
1010 LET C=X(1)
1020 IF C<X(2) THEN LET C=X(2)
1030 FOR I=1 TO 2
1040 IF X(I)/C=INT (X(I)/C) THEN NEXT I: GOTO 1060
1050 LET C=C-1:GOTO 1030
1060 LET A=X(1)/C
1070 LET B=X(2)*A
1080 RETURN
2000 REM ***FINAL DEL PROGRAMA***
2010 PRINT :PRINT :PRINT :PRINT
2020 PRINT "PULSE CUALQUIER TECLA"
2030 PRINT "PARA CONTINUAR."
2040 PRINT "S PARA SALIR DEL PROGRAMA."
2050 INPUT Z$:IF Z$="S" OR Z$="s" THEN END
2060 RETURN
```

Modificaciones para otros equipos

El programa es válido para Spectrum. En el caso de los demás equipos habrá que realizar las siguientes variaciones:

AMSTRAD

2050 INPUT Z$:IF Z$="S" OR Z$="s" THEN END

COMMODORE

20 PRINT CHR$(147)
910 PRINT CHR$(147):INPUT "NUM.FRACCION 1:";N1
2050 INPUT Z$:IF Z$="S" OR Z$="s" THEN END

MSX

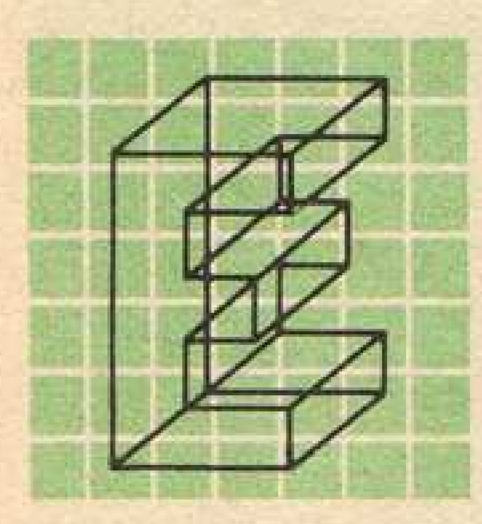
2050 INPUT Z$:IF Z$="S" OR Z$="s" THEN END

IBM

2050 INPUT Z$:IF Z$="S" OR Z$="s" THEN END

PEQUEÑA HISTORIA DE LA INFORMATICA

El ábaco



L ábaco es quizá el primer instrumento para contar creado por el hombre. Cuándo apareció y dónde no está muy claro, pero en casi todas las civilizaciones existe algún tipo de ábaco, desde la China a la precolombina. Desde luego, en Asia ha tenido y tiene gran importancia. Egipcios, babilonios, indios, etc., conocían y utilizaban distintos tipos de ábaco para contar. En cuanto a su edad, cifrémosla al menos en unos dos mil quinientos años, pero probablemente sea más antigua aún.

Existen muchos modelos de ábacos. Unos consisten básicamente en una bandeja con una serie de ranuras sobre las que se colocan las cuentas, siguiendo un determinado orden. (Aquí aparecen los distintos sistemas de numeración utilizados por cada cultura, de los que hablaremos más adelante en la obra.) Otros consisten en un bastidor, sobre el que se sujetan varias varillas, donde van insertadas las cuentas. Entre uno y otro tipo de ábaco existen innumerables variaciones (número de ranuras-varillas, número de cuentas por ranura-varilla, etc.).

Volvamos, sin embargo, al principio de la historia, o mejor prehistoria de este instrumento.

Los hombres de las cavernas no utilizaban ábaco. Sin embargo, fueron facilitando poco a poco su aparición. Cuando necesitaban anotar el número de mamuts que había cazado la pequeña tribu que vivía en la caverna, se limitaban a hacer muescas en la roca. Las anotaciones resultan muy claras, pero son absolutamente fijas. No es fácil arrancar un trozo de roca para mostrar al amigo de la gruta de enfrente que a pesar de haber cazado bastantes fieras, ya les quedan pocas de reserva. Otro sistema menos fijo, pero más inmediato, consiste en agrupar dientes, piedras o huesecillos en hileras, e ir contando el número de hileras y el número de piedras. Estos métodos

¿Sabía usted que...

Raimundo Lulio, gran filósofo medieval, sólo decidió dedicarse a la ciencia a partir de sus cuarenta años?

Como muchos otros hombres insignes, Lulio fue un gran vividor hasta que "sentó la cabeza". Nació en Palma de Mallorca en 1232. Su familia era noble, y en su juventud fue paje de Jaime I el Conquistador. Se casó muy joven y tuvo dos hijos, pero llevó una vida disipada, amando a mujeres de toda condición. La leyenda de su conversión recuerda la de san Francisco de Borja y la reina Isabel, unos siglos después. Andaba Raimundo persiguiendo a una señora casada, que al parecer no cedía a sus intenciones. Le enviaba todo tipo de cartas y poesías amorosas sin obtener resultado. Un día, la señora de sus sueños, cansada de su persecución, le mostró uno de sus senos, destrozado por la enfermedad. Lulio quedó espantado por servir a intereses de vida tan efímera, y decidió retirarse al monte a orar.

En esta montaña, que hoy en día puede visitarse en Mallorca, Lulio, después de muchos días de ayuno, recibió la inspiración divina y escribió su *Ars Magna*.

tan rudimentarios existen en nuestros días, por ejemplo, en Africa, donde en ciertas tribus las cuentas las realizan dos personas, una va contando unidades y la otra otras cantidades, dependiendo del sistema elegido. (En España, ciertos jugadores de cartas cuentan sus tantos utilizando garbanzos generalmente. Unos cuentan las unidades y otros los múltiplos de cinco garbanzos, o amarracos.)

Muchos historiadores consideran, sin embargo, a Egipto como cuna del ábaco. Los egipcios del primer milenio antes de Cristo contaban seguramente de forma semejante a los primeros hombres de las cavernas, pero como desde luego tenían más arena que cuevas, y éstas las utilizaban con cierta frecuencia para esconder las momias de sus faraones junto con algunos de los "recuerdos" que constituían su ajuar mortuorio, era algo más práctico trazar rayas en la arena y luego llenarlas con guijarros.

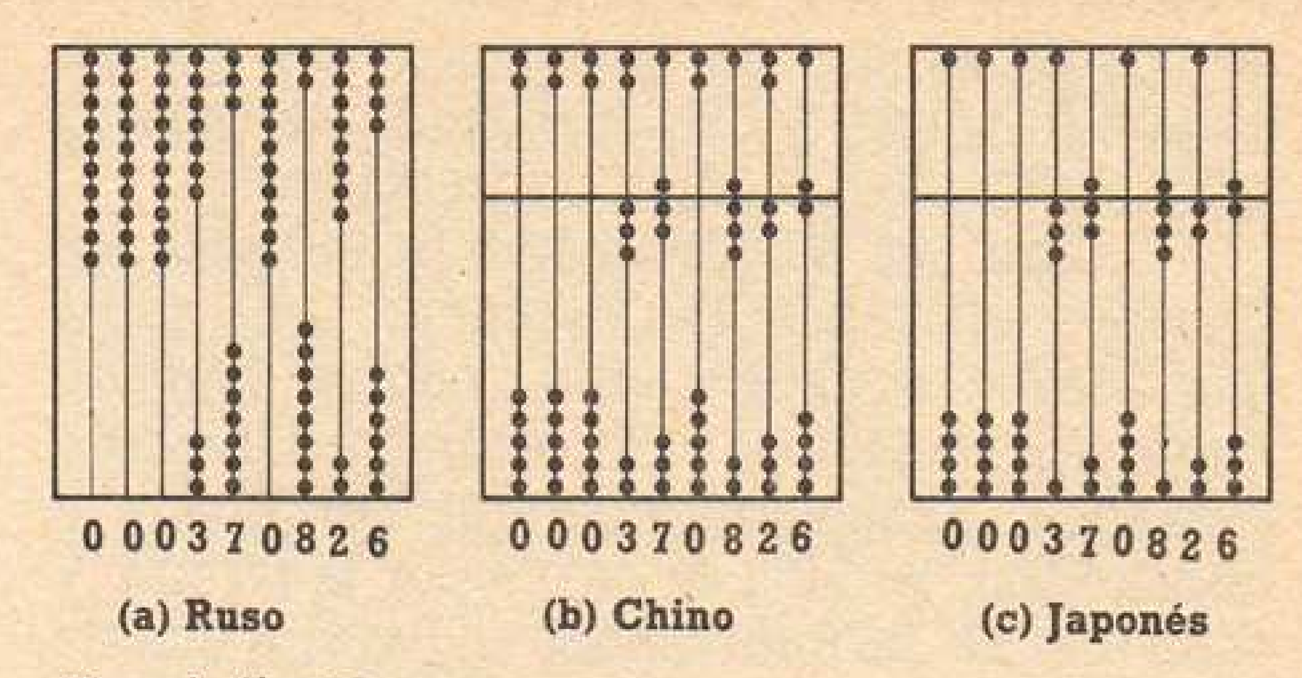
Generalmente, trazaban tres rayas y comenzaban a depositar las piedras en la marca de la derecha, siguiendo ordenadamente. Si necesitaban representar más elementos, comenzaban a añadir más piedras, columna por columna, siempre comenzando por la de la derecha, y transportándolas cuando fuera necesario.

Evidentemente, los egipcios habían dado con un sistema bastante ordenado de contar, aunque, desde luego, no era "portátil". (El problema era el mismo que el de los primeros hombres de las cavernas.) Resultó, por tanto, muy claro que era mucho más práctico hacer las muescas sobre bandejas o tablas, y depositar sobre ellas los guijarros.

Sobre este ábaco primitivo existían ciertas variaciones, como tablas sencillamente recubiertas de arena, sobre las que se podía marcar o dibujar. (La palabra fenicia "abak" describe exactamente lo anterior: superficie plana, cubierta de arena para dibujar y anotar.)

Naturalmente, la siguiente modificación se "cae por su propio peso": para evitar que los guijarros se caigan o se muevan de lugar es mucho más práctico que estén insertados dentro de una guía.

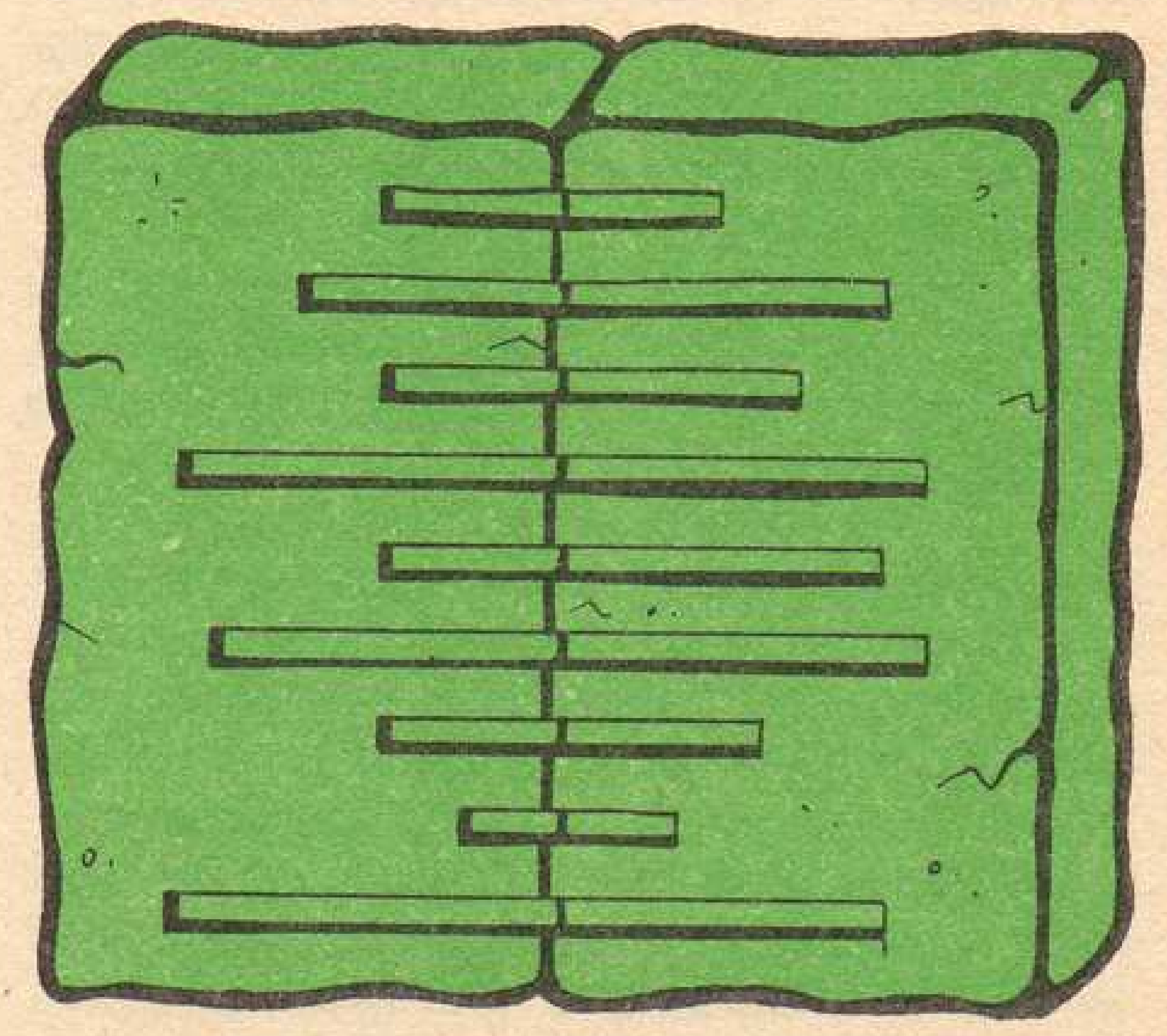
El ábaco greco-romano es también muy antiguo. Se cree incluso que es el precursor del ábaco chino. Los romanos eran unos hombres eminentemente pragmáticos. Sus edificaciones, por lo general grandiosas, siempre tenían fines prácticos. Los monumentos de su época que han llegado hasta nosotros así lo demuestran: entre ellos podemos enumerar las calzadas romanas, acueductos, circos, hipódromos, puentes, baños, etc. Si comparamos este tipo de arquitectura con la medieval, observaremos que, aparte de castillos, iglesias y, en menor proporción, palacios (de cuya utilidad no dudamos) apenas si quedan edificaciones cuyo fin inmediato sea la resolución de algún problema concreto.



Tipos de ábacos.

La practicidad de los romanos también se vio plasmada en sus leyes, normas, etc. Y, naturalmente, utilizaban ábacos. El ábaco greco-romano consistía simplemente en una tabla con un cierto número de marcas paralelas. En ella se comenzaban a contar las cuentas de izquierda a derecha, al contrario que ocurría con el ábaco egipcio. Más adelante, el ábaco greco-romano tomó la forma que tiene hoy en día, y que, curiosamente, ha ido tomando en las distintas culturas donde se ha utilizado: un bastidor sobre el que se fijan unas cuerdas o varillas por las que se pueden deslizar unas bolas pequeñas, que son las que van representando las cantidades.

Plinio, Cicerón y Juvenal nos hablan de este instrumento en sus escritos. Por ellos po-


Taja romana.

demos saber que se perfeccionaron mucho, llegando incluso a disponer de otras ranuras adicionales para la suma de fracciones.

Para sus transacciones financieras los romanos utilizaban generalmente una tabla que llevaba una acanaladura central, llamada **tanja**. Las anotaciones se llevaban a cabo atravesando la acanaladura central. Terminada la transacción, la tanja se partía por la ranura, y cada uno de los interesados se llevaba una de las partes. Para llevar las cuentas de sus negocios utilizaban tablillas de cera, mármol o madera.

Los chinos antes de usar el ábaco que hoy manejan con inusitada maestría, utilizaban unas tablillas parecidas a la tarja romana, llamadas **Suan Pan**.

Sin embargo, llegaron a perfeccionarlo enormemente, y en nuestros días, cuarenta siglos después de su invención, todo niño chino maneja el ábaco mejor que un niño europeo el compás; es un elemento cotidiano, se utiliza para todo. Es más, existen innumerables concursos y competiciones por todo el país.

El ábaco chino se basa en un sistema bipenta (dos grupos de cinco). Esencialmente consiste en el bastidor de que hemos hablado antes, con unas varillas metálicas en las que van insertadas las cuentas. El Suan-Pan está dividido en dos zonas denominadas "cielo" y "tierra". En la zona de "tierra" existen cinco cuentas por varilla, y en la de "cielo", dos. Así, una bola de tierra tiene un valor 1, mientras que la de cielo tiene valor 5. Para sumar valores, las cuentas deben moverse hacia la separación central entre el "cielo" y la "tierra", y para restar, a la inversa. En tiempos de Confucio apareció en China un tipo de ábaco que utilizaba varas de bambú en lugar de cuentas. Este diseño de ábaco es el que se ha estado utilizando hasta hace muy pocos años en la península de Corea.

Se cree que el ábaco chino tiene procedencia romana. Desde luego, el ábaco pasó al Japón desde China, aunque eso ocurrió hacia los siglos XV o XVI.

Como acabamos de mencionar, el ábaco pasó al Japón muy tarde (hemos de pensar que los primeros ábacos, que probablemente fueron egipcios, databan del primer milenio antes de Cristo). Sin embargo, tuvo un desarrollo enorme. Uno de los prohombres japoneses interesados en el instrumento fue Seki Kowa, matemático que desarrolló un sistema de cálculo infinitesimal. Curiosamente, Seki Kowa nació el mismo año que Newton.

¿Sabía usted que...

Raimundo Lulio, filósofo autor del *Ars Magna*, fue el creador de unos diagramas lógicos que al aplicarlos podían descubrir verdades no matemáticas?

También diseñó una especie de máquina mecánica primitiva que facilitaba la ejecución de operaciones lógicas.

Leibniz era un gran admirador de Lulio, aunque lo criticó en muchas ocasiones. Su obra *Dissertio de Arte Combinatoria*, publicada en 1666, constituye su principal crítica de las teorías de Lulio.

El **sorobán**, o ábaco japonés, difiere del chino en que elimina una de las dos cuentas del "cielo" y otra de las cinco cuentas de "tierra", que quedan así reducidas a cuatro. Este cambio supuso una mejora evidente en simplicidad.

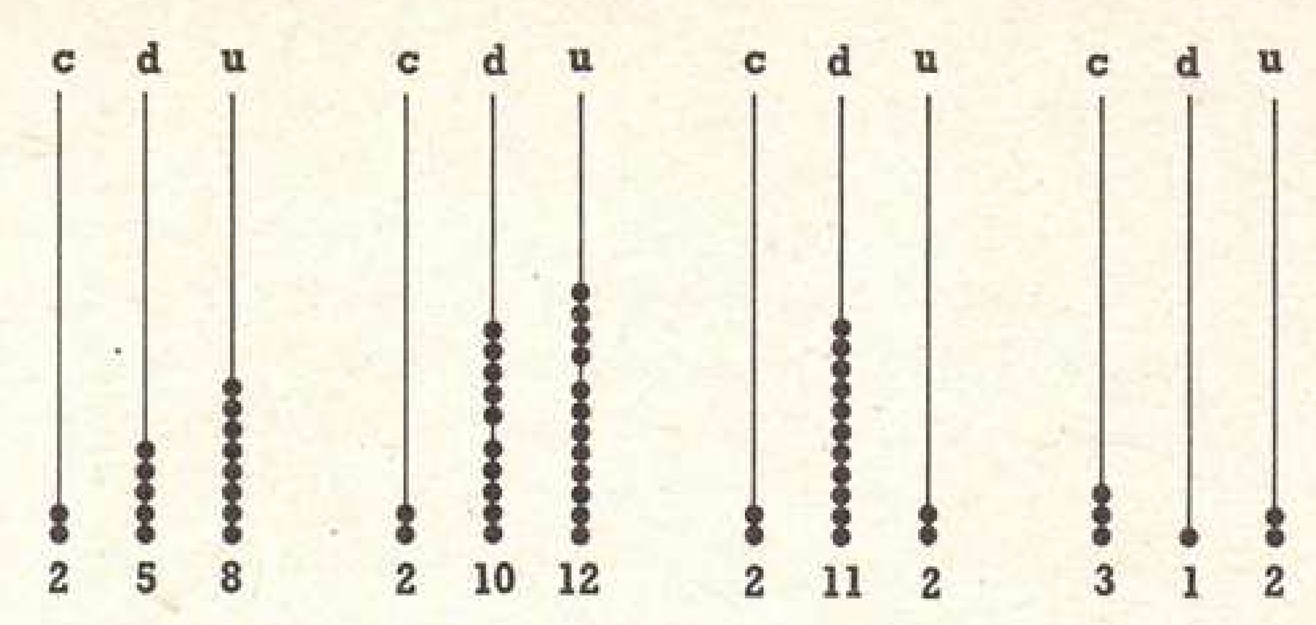
El desarrollo del ábaco en el Japón ha sido impresionante. En ese país existe un instituto de investigación del ábaco, que desarrolla métodos, patrocina concursos, etc.

En 1946 un experto en máquinas calculadoras del ejército americano. T. N. Wood, seleccionado entre muchos otros hombres, se decidió a competir con una calculadora eléctrica contra un japonés llamado Matzuzaki, que manejaba un ábaco. La competición constaba de cinco ejercicios de cálculo y se valoraba exactitud y velocidad. El vencedor fue el japonés y el tanteo fue de cuatro a uno.

La maestría de los japoneses con el ábaco es increíble. Realmente las sumas y restas se realizan con gran facilidad, pero las multiplicaciones y divisiones son muy reiterativas y los cálculos intermedios son muy difíciles de comprobar. A pesar de ello, los japoneses consiguen velocidades de cálculo impensables con un instrumento de cálculo tan rudimentario.

Saltando de continente y también en el siglo XV, los españoles llegan a América y observan que los indios también contaban con un ábaco, que era prácticamente igual a los desarrollados en otras civilizaciones. Tenía el mismo bastidor, las varillas, pero en este caso las pequeñas piedras o cuentas se iban introduciendo en las varillas, es decir, el bastidor estaba abierto.

En la Edad Media, en Europa, las casas nobles poseían sus propias tablas de cálculo. Naturalmente, no eran ábacos de elaboración



Suma de 258 más 54 en un ábaco primitivo.

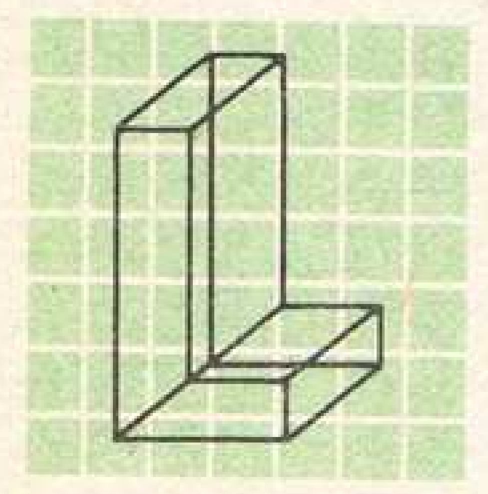
sencilla (los artesanos hábiles no escaseaban en aquellos tiempos). Era frecuente, pues, que las tablas fueran lujosas, con bolas talladas en materiales nobles, o fundidas. También era relativamente corriente que llevaran grabado el escudo de la casa. Pasaban de padres a hijos, constituyendo un signo de distinción.

A veces, los tableros de contar eran pesados y voluminosos, y por ello, poco prácticos. (Recuerde el lector que en muchas zonas rurales se sigue llamando "contador" al mostrador de una tienda.) La palabra contador era adecuada, ya que era exactamente eso: el lugar donde el tendero hacía las cuentas, su tablero de contar. Otra costumbre que se fue instaurando poco a poco fue trazar las marcas de contar sobre un paño de lana. Y aquí aparecen los franceses, que llamaban a este paño, y a los paños que cubrían las tablas, "bure". De ese paño deriva la palabra francesa "bureau", la española burócrata, etc.

Sin embargo, hacia el siglo XV (al contrario de lo que sucedió en el Japón) los ábacos comenzaron a decaer. La aparición del sistema de numeración arábiga tuvo mucho que ver con su desaparición.

Hoy, en pleno siglo XX, en muchos países asiáticos, fabricantes de calculadoras rapidísimas, el ábaco sigue teniendo vigencia. Y en nuestra civilización quedan vestigios de estos sistemas rudimentarios de contar. El backgammon, por ejemplo, es un juego que deriva directamente del ábaco.

Criptología informática



A criptología es la ciencia de la ocultación de la información. Normalmente la parte más conocida de la criptología es la criptografía, o ciencia de la escritura ("grafía") oculta o secreta ("cripto"). Se desarrollan en Criptografía técnicas y algoritmos para "cifrar" (u "ocultar") los mensajes y poder transmitirlos o almacenarlos de un modo más seguro; en el punto de destino es necesario disponer de los mismos algoritmos y técnicas (clave de cifrado) para volver a obtener el mensaje o datos en una manera inteligible (se llama mensaje "claro").

Siempre se han utilizado técnicas de cifrado para el envío seguro de mensajes. Con frecuencia (sobre todo en épocas de civilizaciones primitivas) se han utilizado medios mecánicos o materiales de ocultación: texto escrito a lo largo de un bastón en un pergamino enrollado en dicho bastón —o escítalo— y que sólo podía ser leído fácilmente con ayuda de otro bastón igual en longitud y diámetro; texto repartido en varias piezas trasladadas por diferentes procedimientos y que sólo en el punto de destino se unirán de acuedo con una determinada clave para componer el texto completo; o, incluso, mensaje escrito en la cabeza afeitada de un esclavo (al que se envió cuando ya tenía el pelo crecido) y que sólo pudo ser leído volviendo a afeitar la cabeza al esclavo. Sin embargo, el método más utilizado tradicionalmente ha sido el de cambio de simbología (o sustitución): en efecto, el procedimiento más inmediato y rápido (tanto en el momento de cifrado como en el de descifrado) es la sustitución de cada letra del alfabeto y número por un símbolo prefijado (o varios) que puede ser un símbolo diseñado al efecto u otra letra (o número). Aun así, se puede hacer el proceso de sustitución bien de acuerdo con una clave fija "anárquica" (código de sustitución permanente) o bien de acuerdo con una norma: por ejemplo, sustituir cada letra por su "simétrica" en el alfabeto (la c, tercera por delante, se sustituye por la x, tercera contando desde el final) o por la que está tres posiciones a su derecha (la c se sustituye por la f), etc. E incluso, se sustituye cada símbolo por el que está "n" posiciones a su derecha, donde "n" viene dado por el número de letras de la palabra enviada.

Incluso se puede complicar el esquema y utilizar varios alfabetos (sistemas polialfabéticos) o hacer las sustituciones en función de la frecuencia de los diferentes caracteres dentro del alfabeto. En ocasiones se utiliza una sustitución digráfica (se sustituyen los caracteres del texto original de dos en dos).

Sin embargo, en los métodos de sustitución permanece subyacente un problema: la "forma" del texto (distribución de signos con sentido y de espacios para formar las palabras) permanece y puede ser observada. Para obviar esta dificultad se han utilizado numerosos procedimientos de cambio de posición de los símbolos (transposición), además de la técnica de rellenado (sustitución de los espacios en blanco por un símbolo adicional).

Los procedimientos de transposición pueden ser simples, como la inversión del orden de lectura o el cifrado por líneas (se coloca del mensaje —previa eliminación de los

espacios entre palabras— en varias líneas y se lee por columnas) o en diente de sierra (se coloca en varias columnas desplazadas y se lee por líneas); o bien sistemas más sofisticados para determinar el orden de selección de las letras del texto: sistema matricial (en el que se colocan las letras en forma de matriz y se seleccionan posteriormente según diferentes algoritmos de localización); sistema por columnas a partir de la llave (el orden alfabético de las letras de la "palabra llave" indican el orden en que han de ser seleccionadas las columnas en que se ha escrito el texto —tantas columnas como letras tiene la "palabra llave"—); método matricial de Cardano (se seleccionan las letras que indican unos cuadros marcados en una cartulina, se gira la cartulina y se vuelven a seleccionar, etc.).

La introducción de los ordenadores en la criptografía añade dos posibilidades, básicamente: por un lado, cualquier proceso de cifrado/descifrado se hace más rápida y fiablemente con un ordenador electrónico que normalmente o mediante una máquina (lo que supone que se pueden definir algoritmos de cifrado más complejos o que se pueden repetir más veces algunos pasos básicos de transposición o sustitución); por otro lado, se pueden utilizar las propias capacidades del ordenador para definir los algoritmos criptográficos.

En esta segunda línea se utilizan actualmente sistemas de cifrado basados en la suma y resta binarias, en la multiplicación y división, en el OR EXCLUSIVO, equivalencia lógica y negación. La idea básica de todos estos esquemas de cifrado es que se puede "operar" el texto claro con la clave mediante cualquiera de esas operaciones para dar un texto cifrado (en una primera etapa); posteriormente se pueden efectuar transposiciones o alguna sustitución adicional. Así, la palabra TRES puede estar representada por los números 20, 18, 05, 19 (según su posición en el alfabeto) y sumada con la llave CLAVE (igual a 03, 12, 01, 22, 05) dará 0332192724 (puesto que 201805 + 0312012205 = 0332192724), que corresponde a CFSAX (donde se ha restado 26 de los números que rebasan dicha cifra; por ejemplo, el segundo grupo, 32, se toma como 6 = F). Sin embargo, dicha palabra TRES operada mediante equivalencia lógica con la llave CLAVE, dará CHDVA (habiendo operado en binario los valores hexadecimales de cada número de la tabla —20180519 para TRES y 0312012205 para CLAVE— y restando 9 a cada código hexadecimal de la A a la F y, por último, restando 26 o sus múltiplos de los valores decimales resultantes).

La gran ventaja del ordenador en estos métodos es que opera "en su propio terreno" (es decir, mediante representaciones próximas a su modo nativo = el 0 y el 1) y, por tanto, opera con suma rapidez.

Basados en estos procesos básicos de cifra se han desarrollado algunos algoritmos

METODOS CRIPTOGRAFICOS

A. METODOS CLASICOS

- A.1. Sustitución.
  - A.1.1. Monoalfabéticos.
    - De César.
    - Variante con inversión del alfabeto.
    - Variante con conversión intermedia en números.
    - Método de matriz.
    - Mediante palabra clave.
  - A.1.2. Polialfabéticos.
    - Tabla Vigenere.
    - Tabla con clave.
  - A.1.3. Sustitución digráfica.
- A.2. Transposición.
  - A.2.1. Inversión del orden.
  - A.2.2. Transposición por líneas.
    - Lectura por columnas.
    - De diente de sierra.
  - A.2.3. Matriciales.
  - A.2.4. Por columnas a partir de la llave.
  - A.2.5. De Cardano.
- A.3. Sistemas híbridos.

B. METODOS CON ORDENADOR.

- B.1. Esquemas básicos.
  - Suma y resta.
  - Producto y división.
  - Operaciones lógicas.
  - Esquemas matriciales.
- B.2. Algoritmos sofisticados.
  - B.2.1. De llave secreta.
    - Llave en memoria.
    - De llave infinita.
    - D.E.S.
  - B.2.2. De llave pública.
    - R.S.A.
    - De la "mochila".

criptográficos más sofisticados: método de "la llave en memoria" (utiliza dos llaves y una dirección de la memoria del ordenador; opera con el OR EXCLUSIVO va modificando las direcciones de memoria con las que opera); método de "la llave infinita" (opera con el OR EXCLUSIVO también, pero utiliza como llave una serie de números aleatorios obtenidos a partir de una determinada "semilla" o valor inicial, que es la verdadera llave del método); e incluso un proceso más complicado (sistema D.E.S.), en que intervienen varias transposiciones fijas y sustituciones controladas por llave (método desarrollado originalmente por los expertos de IBM y que, posteriormente, fue adoptado por la Agencia Nacional de Seguridad de los EE.UU.).

Además, existen algunos sistemas llamados de llave pública, de los que se conoce perfectamente el algoritmo de cifrado y cada usuario debe poner los números de clave: en el sistema RSA (utilizado por el Departamento de Defensa de los EE.UU.) se parte de dos números enteros (e y n), se distribuye el mensaje en bloques y se obtiene el resto de dividir el mensaje anterior entre n. El método de Merkle y Hellman se basa en el problema llamado "de la mochila": dados un conjunto de elementos de longitudes l_1, l_2, ... l_n y dada la longitud total a obtener "e", elegir un subconjunto de longitudes "l_i" tales que sumadas den el total "l"; mediante este método se genera un conjunto de números aleatorios comprendidos entre 0 y m–1 (a partir de un conjunto de tres m, n y p, tales que $n. \cdot p = 1 \pmod{m}$), de tal modo que se hacen públicos los números obtenidos pero no los de la llave a partir de la cual se han calculado.

TERMINOLOGIA

GLOSARIO DE TERMINOS UTILIZADOS EN CRIPTOLOGIA INFORMATICA

Algoritmo. Conjunto de operaciones de cálculo desarrolladas en una secuencia preestablecida y conducente (en un número finito de pasos) a la resolución de un problema específico.

Cifra. Se suele designar por este nombre al algoritmo de cifrado u ocultación del mensaje. En ocasiones se denomina cifra al propio proceso de cifrado.

Cifrado. Proceso de ocultación mediante métodos criptográficos. "Texto cifrado" u oculto es el que se obtiene después de aplicar el algoritmo de cifra al mensaje original o mensaje claro.

Claro. Dícese del mensaje inteligible (mensaje original o mensaje resultante del proceso de descifrado), en contraposición al mensaje cifrado.

Clave. Véase LLAVE.

Criptoanálisis. Parte de la criptología centrada en el análisis de los algoritmos de cifra y en el descifrado de mensajes ocultos.

Criptografía. Estudio de la escritura oculta o cifrada de mensajes. Conjunto de técnicas orientadas a este fin.

Criptología. Ciencia que estudia los procedimientos de ocultamiento de datos o mensajes. Suelen distinguirse en ella la criptología y el criptoanálisis.

Des. Data Escryption Standard. Nombre de un código de llave secreta desarrollado por expertos de IBM y adaptado posteriormente por la Agencia Nacional de Seguridad (NSA) y la Oficina Nacional de Estándares (NBS) de los EE.UU.

Descifrado. Proceso de puesta en claro de un mensaje. Proceso inverso a la cifra.

Digráfica, sustitución. Sustitución digráfica es la que se realiza aplicando el proceso de sustitución a los caracteres del mensaje claro tomados de dos en dos.

Equivalencia lógica. Operación de proceso de dos números binarios (dígito a dígito) que da como resultado un 1 si ambos operandos tienen el mismo valor y un 0 si no coinciden, de acuerdo con la siguiente tabla:

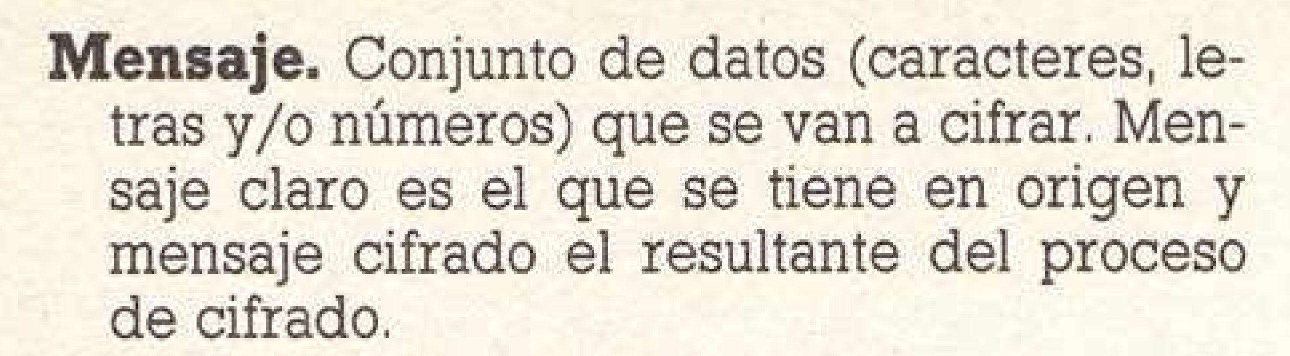
≡	0	1
0	1	0
1	0	1

Hexadecimal. Sistema de numeración en base 16. Es muy cómodo para representar los grupos de 4 bits (ceros o unos) con que trabaja el ordenador. Sus dígitos se representan por los números del 0 al 9 (con el mismo valor que en base 10, desde luego) y las letras A (valor 10) a F (valor 15).

Llave. También llamada clave. Es la combinación (de caracteres normalmente) que sirve de base para la operación del algoritmo de cifrado. Sus elementos indican a veces el orden en que deben ser efectuadas las operaciones y, en otras ocasiones, son utilizados como operandos en alguna operación aritmética o lógica.

Matriz. Conjunto de elementos ordenados en filas y columnas formando un rectángulo de dimensiones dadas.

TERMINOLOGIA



Mensaje. Conjunto de datos (caracteres, letras y/o números) que se van a cifrar. Mensaje claro es el que se tiene en origen y mensaje cifrado el resultante del proceso de cifrado.

Monoalfabéticos. Algoritmos de cifrado que manejan un solo alfabeto.

Or exclusivo. Operación de proceso lógico de dos números binarios (dígito a dígito) que da como resultado un 1 si uno cualquiera de los dos operandos (pero no ambos) es 1 y el otro 0; y toma como resultado un 0 y si ambos operandos son 0 o ambos 1. La tabla correspondiente de la operación es:

$\oplus$	0	1
0	0	1
1	1	0

Polialfabéticos. Dícese de los algoritmos de cifrado que operan con dos o más alfabetos.

RSA. Algoritmo de cifrado ideado por Rivvest, Shamir y Aldeman. Es de llave pública y ha sido seleccionado por el Departamento de Defensa de los EE.UU. para aplicaciones de alta seguridad.

Semilla. Se suele designar así el valor de base utilizado en un proceso de creación de números aleatorios.

Sustitución. Procedimiento básico de cifra consistente en poner un símbolo o carácter en lugar de otro (de acuerdo con un algoritmo prefijado), pero sin cambiar su posición relativa. Normalmente se combina con procesos de transposición.

Transposición. Procedimiento básico de cifra consistente en cambiar el orden de los caracteres, según el proceso que se haya decidido efectuar, independientemente de su sustitución o no.

VOCABULARIO DE INFORMATICA

Bit de signo. En un sistema binario sirve para indicar el signo algebraico de un número. Es el bit situado más a la izquierda de la palabra. Los restantes bits se utilizan para representar la magnitud del número.

Bit menos significativo. Bit de menor peso en la palabra de que se trate. Generalmente es el bit situado más a la derecha. Por ejemplo, en el byte 1001 0101, el 1 de la derecha es el bit menos significativo.

Bits por pulgada (BPI). Sistema para medir la densidad de grabación en los medios magnéticos. Indica el número de bits que se pueden grabar en una pulgada de soporte magnético (discos flexibles, cintas, etc.).

Bits por segundo (BPS). Sistema para medir la velocidad de transmisión de datos. Indica el número de bits transmitidos por cada segundo.

Bit slice (troceamiento por bits). Ciertos dispositivos están organizados de forma que cada circuito integrado procesa únicamente una parte (ciertos bits) del número binario de que se trate. Generalmente son dispositivos en los que la velocidad es un elemento primordial. Normalmente se trata de unidades logicoaritméticas muy sencillas cada una individualmente. Este tipo de dispositivos son los utilizados, por ejemplo, en el proceso de imágenes de visión artificial.

Biyección. Aplicación inyectiva y suprayectiva, es decir, a cada elemento del conjunto original le corresponde uno y sólo uno de los elementos del conjunto imagen.

Blanco (blank). Zona o parte del medio en el que no está registrado ningún carácter.

Bloque. Conjunto de elementos, palabras, caracteres o dígitos que se tratan considerándolos como una unidad.

Colección de registros grabados formando una unidad. Estos bloques pueden contener uno o más registros y están separados por huecos (gaps).

Grupo de bits o dígitos transmitidos formando una unidad. Grupo de caracteres contiguos, grabados como una unidad. Del mismo modo que los bloques de registros, están separados por huecos (gaps).

Bloques, diagrama de. Representación simplificada de un dispositivo o de un proceso en el que las partes esenciales están representadas por figuras geométricas (rectángulos y rombos normalmente) que indican tanto los valores como las funciones y las relaciones existentes entre los elementos representados.

Block gap. (Ver bloques, separación entre.)

Bloques, separación entre. Zona del medio magnético (disco o cinta) que se utiliza para indicar el final de un determinado bloque de registros.

Bloque, longitud del. Medida del tamaño del bloque. Normalmente se mide en registros, palabras o caracteres.

Bloques, transferencia de. Proceso por el

cual se transmiten uno o más bloques de datos (siempre que éstos estén ya organizados en bloques).

Booleano. Relativo a un proceso realizado en álgebra de Boole. Relativo a operaciones de Lógica Formal.

Booleana, función. Aplicación del producto cartesiano ("n" veces) de un conjunto en el mismo, establecida de tal modo que la función viene definida a partir de las operaciones del álgebra de Boole (suma producto y complementación).

Booleana, suma. Idéntico a OR, 0 o disyunción. En una álgebra binaria toma valor uno cuando cualquiera de los dos "sumandos" valen uno.

Booleano, operador. En Informática se designa con este nombre a un operador lógico en el cual operandos y resultado sólo pueden tomar dos valores.

Bootstrap (inicialización). Dispositivo de arranque. Programa o procedimiento o rutina para poner en marcha el ordenador de que se trate, por sí mismo.

Borrado. Eliminación de un dato o registro. También puede decirse del cambio de datos o registros, al grabarse sobre ellos. Liberación de una zona de la memoria.

Borrado de un fichero de trabajo. Eliminación de un fichero transitorio. Eliminación de un fichero utilizado para almacenar datos producidos como salida de un programa. Realmente, los datos no son borrados, sencillamente se elimina la referencia a ese fichero en el índice correspondiente.

Bottom up (de abajo a arriba). Término que se utiliza en Informática para designar programas creados sin consideraciones de bloques. Las instrucciones se escriben detalladamente una tras otra.

BPI. (Ver Bits por pulgada.)

BPS. (Ver Bits por segundo.) A veces, si se especifica con más claridad, puede referirse a bytes.

Break. Interrupción de un programa. Tecla que interrumpe el programa.

Break point. Punto de una rutina especificado mediante un dígito, condición o instrucción en la que ésta puede ser interrumpida por una intervención externa o en la que la propia rutina se detiene para realizar comprobaciones o controles.

BS. El carácter de retroceso backspace. Tecla backspace del teclado del ordenador que hace desplazarse al cursor una posición hacia atrás.

Bucle. Término que indica un grupo de instrucciones de un programa que deben repetirse cierto número de veces. Cada lenguaje tiene unas instrucciones específicas para la programación de bucles.

Buffer. Area de la memoria (parte de la memoria principal del ordenador o memoria dedicada exclusivamente a este fin), que se destina a almacenamiento temporal de datos en la transmisión a un periférico. Habitualmente suele haber dos memorias intermedias o buffer, una de entrada y otra de salida.
Circuito destinado a evitar las dificultades debidas a la diferencia de velocidad de trabajo del "emisor" de los datos y del receptor. En un circuito que almacena temporalmente los datos o retarda la transmisión para sincronizar las velocidades.

Burótica. Se refiere a aquellas aplicaciones informáticas relacionadas con el trabajo de oficina. Deriva del francés "bureau informatique", oficina informática.

Bus. Conjunto de líneas de comunicación entre dos dispositivos. A través del bus, se pueden transmitir tantas señales como líneas posea éste.

Bus de datos. Bus destinado exclusivamente a la transmisión de datos. Debe disponer del mismo número de líneas como bits tenga la palabra que envía o recibe el ordenador.

Bus común de direcciones. Enlace común (bus) dedicado específicamente a transmitir información sobre direcciones. Puede ser un bus dedicado exclusivamente (claramente separado de otros buses), o formar parte de un bus genérico de transmisión de información.